Standardization, Open Source and Innovation Ecosystem

Martin HUSOVEC

Contents

1. Introduction

Open source and standardization can be described as two 'stewards of innovation' (Kapos, 2017). Although practically two different ecosystems with diverging set of rules and objectives, they, however, meet in their purpose to push the frontier of innovation. The latest technological developments are increasingly pushing firms and individuals participating in these ecosystems to work more closely together. However, under whose rules? And with what effects?

At the first look, open source and standardization are almost odd to compare. The former deals with products, the latter with basic specifications on which those products are based. Firms cooperate on specifications through standardization, by creating a common level-playing field, and then compete on product implementations. After a deeper look, however, it appears odd *not* to compare them. In the recent OMA Survey, from 419 respondents, 70 % were of the view that open standards play 'an entirely different' role than open source; only 10 % participants disagreed and the remaining 20 % said to be 'neutral' towards the claim. If the goals are so clearly different, why would this be?

The argument put forward in this article is that interactions of open source and standardization activities take place in four basic scenarios which should determine the answers to the questions posed. Depending on the scenario, open source and standardization activities can be said to play complementary, overlapping or, exceptionally, even identical role. These dynamics, more than anything else, influence the types of conflicts and their potential treatments.

[W3C when switching to RF in Jorge + IETF re: future]

2. Two Ecosystems

The driving force behind convergence of open source and standardization ecosystems is the fact that ever more features are being implemented by software instead of hardware. In the area of networking technologies, for instance, a big part of development today happens in open source projects.[1] Virtual private networks virtualize entire classes of network node functions into building blocks that may connect, or chain together, to create communication services.[2] This allows to substitute custom hardware appliances by software development. This trend pushes many SDOs, such as IETF and ETSI, towards closer collaboration with open source projects.

[software standards + changes + areas]

[first some basics, and linking]

2.1 Standardization

In the area of ICT, standardization is a process of specification of the technical design features,[3] which allow different products of multiple vendors to work together. Traditional view has been that standards

[1] See https://www.ietf.org/archive/id/draft-arkko-ietf-trends-and-observations-00.txt
[2] See https://en.wikipedia.org/wiki/Network_function_virtualization
[3] ITU-T defines standard as XXX

and firm innovation contradict (Blind 2009, p. 3). Therefore network externalities, efficiency in the supply chains, economies of scale or reduction of transactions costs (Swan (2000)) would be cited as more obvious economic benefits. However, increasingly, the catalyst role of standards is getting more attention (Blind 2009). This is because in many modern industries, standardization is becoming an essential prerequisite for the diffusion of innovative technology (Baron and Schmidt 2016). Although some still argue that standardization in ICT is mainly a tool of technological coordination (Egyedi 2000), looking at projects like 3GPP (Lopez-Berzosa and Gawer 2014), it seems equally possible that they also serves as a platform for Chesbrough's open innovation (Chesbrough 2003). As also pointed out by Blind, besides selection and codification of knowledge in standards, 'an exchange of tacit knowledge takes place during the standardization process' (Blind 2009, p. 9). Hence standards facilitate finding of innovations as well as their follow-on diffusion.

The European Commission (2014) distinguishes SDOs of three broad categories: (1) those that are formally recognized by governments (e.g. ISO, IEC, ITU, ETSI), (2) "quasi-formal" groups that are typically large and well-organized and share many of the characteristics of the first group, but lack official governmental recognition (e.g. IEEE, IETF, W3C), and (3) smaller, privately-organized consortia (e.g., Bluetooth SIG). The governance structure of these SDOs differs significantly because, among other things, they are subject to different external constrains and models of gaining legitimacy.[4] Especially in more formal SDOs, membership and ability to approve standards can be sometimes restricted. However, even in such SDOs, participation in the technical committees is open to all materially interested parties.[5] Quasi-formal SDOs are more likely to have unrestricted membership. In all these organizations, standards are developed in the technical committees, which adopt them by consensus. The resulting documents then have to be usually subsequently approved by further organs of the SDO, e.g. its membership, director or board of directors. The process of drafting standards in the technical committees is guided and regulated by number of SDO policies - the rules on intellectual property rights (IPR) being one of the most crucial.

Copyright policies, focusing on standard documentation, used to be less important for SDOs,[6] with the exception of some SDOs. These, very often formal SDOs, often charge for access to the standardization documents in order to support its work (e.g. ISO, national SDOs, IEEE). The copyright policies are becoming much more important in the area of software standards, such as those developed by IETF, OASIS or W3C, because the code is becoming a way of how to formulate specifications themselves. In the SDOs where no explicit policies exist for software specifications (e.g. IEEE), the usual rules regarding specifications would arguably apply.[7] However, some SDOs adopted also more explicit copyright licensing policies for software specifications (e.g. ETSI,[8] OASIS,[9] IETF[10], or W3C[11]). [What flavors]

[4] REF to JRC study
[5] JRC cite
[6] Lemley (2002): "Thirteen of the thirty-six SSOs with policies applied the policy only to patents. Many other organizations applied different rules to patents and copyrights" + "Another ten SSOs had different rules for patents and copyrights, permitting reasonable and"nondiscriminatory licensing for patents but requiring assignment or royalty-free licensing of copyrights. This is the inverse of ISO's rule."
[7] Jingze, p. 2
[8] See http://www.etsi.org/images/files/IPR/etsi-ipr-policy.pdf (Section 9)
[9] See https://www.oasis-open.org/policies-guidelines/open-repositories
[10] See https://trustee.ietf.org/documents/IETF-TLP-5_001.html
[11] See https://www.w3.org/Consortium/Legal/2015/copyright-software-and-document

To facilitate the diffusion of technologies, SDOs adopt policies which regulate situations when standards include patent rights of its participants. The patent policies usually require disclosure of standard-essential patents and specify basic terms for their availability to other market players. The standard documentation describing specifications is often copyright protected[12] and, depending on the industry, presence of patented technologies can be significant.[13]

In the last two decades, many SDOs adopted rules requiring commitments to license standard essential patents (SEPs) on specified, most often FRAND terms (Baron and Spulber, 2015). The exact meaning of such commitments was subject to extensive debate before the courts and in the literature (Pentheroudakis and Baron (2017), NAS (2013), Bekkers and Updegrove (2012) and ABA (2007)). They do not constitute a license, but only a form of patent pledge.[14] To be sure, name 'FRAND', similarly as 'open source', is better viewed as one category rather than single set of expectations. The exact wording, and thus definitions and permissions, may differ from SDO to SDO,[15] though the variability is usually less significant as in case of different open source licenses. The most widely adopted wording of FRAND commitment is a 'Common Patent Policy for ITU-T/ITU-R/ISO/IEC'.[16] This FRAND commitment only obliges the patent holder to be willing to negotiate licenses 'with other parties on a non-discriminatory basis on reasonable terms and conditions', including by declaring willingness to engage in royalty-free licensing.[17] Such negotiations then take place outside of an SDO. Some SDOs specify additional features of such commitment, such as define how royalties should be calculated (IEEE)[18] or ask SEP owners to specify their maximum future fees or royalties and the most restrictive license terms for licenses (VITA).[19] Instead of simply requiring FRAND commitment from participants, some SDOs require royalty-free licensing (W3C)[20] or provide a broader choice of licensing options (OASIS).[21] IETF is one of the few SDOs that has no FRAND-commitment and relies only on extensive disclosure policy.[22]

2.2 Open source

Open source is a prototype of open innovation. It allows unaffiliated firms or individuals to develop products in a collaborative process. Its popularity is best illustrated by the fact that all major technology firms are today involved in its production. Open source is so omnipresent that a Samsung representative

[12] (OLG Hamburg)
[13] REF
[14] Jorge bok
[15] IEEE's definition
[16] See https://www.itu.int/en/ITU-T/ipr/Pages/policy.aspx
[17] Ibid.
[18] See http://standards.ieee.org/develop/policies/bylaws/sect6-7.html
[19] See https://www.vita.com/Disclosure
[20] See https://www.w3.org/Consortium/Patent-Policy-20040205/
[21] See https://www.oasis-open.org/policies-guidelines/ipr
[22] See https://www.ietf.org/rfc/rfc8179.txt For discussion of history of this approach see Contreras, Jorge L., A Tale of Two Layers: Patents, Standardization, and the Internet (August 30, 2016). Denver University Law Review, Vol. 93, No. 4, 2016; University of Utah College of Law Research Paper No. 177. Available at SSRN: https://ssrn.com/abstract=2832538

recently exclaimed that: "Today, you can't build a product without using Open Source" [23] It is therefore no surprise that standardization clients greatly overlap with participants in open source projects.

Although there are many definitions of open source, the most authoritative is one maintained by the Open Source Initiative (OSI), a private certification authority. Its definition is composed of 10 criteria,[24] such as free distribution, availability of source code, freedom to carry out modifications and non-discrimination. As explained by Shane Curcuru, 'OSI-approval is not a magic stamp; however it does show licenses that are so widely used — and reviewed by lawyers — that there is seen as less risk to everyone else in consuming software under an OSI license.'[25] In order to gain an approval, each license is scrutinized against the set of 10 criteria. In the context of co-existence with FRAND commitments, in the particular the requirement of free distribution often leads to problems.[26] OSI-approved licenses do not allow royalty-bearing licenses and therefore isn't always favored by some type of stakeholders.

Therefore the main organizing principle of any open source project, which influences the surrounding entire ecosystem, is its license. The main problem of the of the open source world is that there is great proliferation of licenses. Only OSI certified 83 licenses to this day,[27] and there are many other licenses that don't qualify despite calling themselves 'open source', or would qualify, but are not yet certified.[28]

An open source license may not only differ in terms, but also in its coverage. We can observe three categories of licenses: (1) copyright-only licenses, where the scope of permissions is explicitly limited to copyrights (e.g. XX); (2) copyright and patent license, where the scope of permissions is explicitly extends to both copyright and patents (e.g. Apache2, GPLv3); and (3) in between licenses, where the copyright permissions are explicit, but patent grants are not articulated and thus subject to ongoing controversy (e.g. BSD, MIT, GPLv2).[29]

In terms of governance, open source projects may be supported by a separate legal entity, but they don't have to be. For instance, world's most famous project, Linux, wasn't hosted by any organization between 1991 and 2000. Today, it is supported by the Linux Foundation. Apart from The Linux Foundation, there are few generalist organizations, such as Apache Software Foundation or The Eclipse Foundation that host multiple projects by providing them with organizational, legal, and financial support. At the same time, we also observe number of project-specific foundations, such as The Document Foundation hosting LibreOffice, Open Course Ware Consortium hosting Open Course Ware software, or OpenStack Foundation hosting Open Stack cloud operating system.

The default in the open source world is that individual developers or their employers retain their copyright and patents, thus licensing to all the users in the world directly, through the project license. Therefore licensor-licensee relationship is between developers and users, and there is no middle-man. Increasingly, however, foundations, where they exist, step in by aggregating the copyright and patent permissions of its members. Such foundations then ask their contributors to sign contributory licensing agreements (CLAs) with the foundation. For instance, Apache Software Foundation asks its contributors for a specified

[23] http://blogs.wsj.com/cio/2014/05/05/open-source-eating-software-world-samsung/
[24] See https://opensource.org/osd-annotated
[25] See https://medium.com/@shanecurcuru/three-reactions-to-the-facebook-patents-license-b64e6942012b
[26] See the debate regarding broad patent retaliation clause [Luis!]
[27] See the list https://opensource.org/licenses/alphabetical
[28] REASONS for not qualifying + others
[29] Finish guy piece http://www.ifosslr.org/ifosslr/article/view/107/205

'a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable' copyright and patent license.[30] This set-up allows the hosting organizations to re-license the project, if necessary, and more easily enforce the project's license. The downside of CLAs are transaction costs, potential antagonization of the community base of developers and potential abuse by foundations.[31]

Copyright / Patent	**Patent Royalty-allowing**	**Patent Royalty-free**
Permissive	Explicit: [XX] IF not implicit: *BSD, MIT*	Explicit: *Apache v2; BSD-2-Clause-Patent; MPL-2.0; UPL v1* IF implicit:[32] *BSD; MIT*
Copyleft	Explicit: [XX] IF not implicit: *GPLv2*;	Explicit: GPLv3 IF implicit: GPLv2

Apart from the patent grants included in the open source licenses, another important element are so called patent retaliation clauses, which can be (a) weak or (b) strong (e.g. MPL 1.1). Patent retaliation clause is an in-built license mechanisms the purpose of which is to withdraw licensing authorizations towards potential licensees that assert their own patent rights against the licensor. The difference between weak and strong retaliation clauses is in their breath. While weak retaliation clauses are usually limited to patent licenses pertaining to underlying software (e.g. GPLv2), strong clauses are cover broader range of actions (e.g. Apache v2) or terminate even copyright permissions (e.g. MPL 2.0). The best example of this kind is a recent controversy around Facebook's React which was licensed under BSD-3-Clause license, which is OSI-approved, but accompanied by Facebook's own custom-written patent declaration. This declaration effectively meant that suing Facebook over patent rights, even if unrelated to the React project, would automatically revoke Facebook's own royalty-free patent grants to such plaintiff.[33] The controversy surrounding this clause was so strong[34] that Facebook eventually decided to relicense the

[30] See https://www.apache.org/licenses/cla-corporate.txt
[31] See for the discussion > http://ebb.org/bkuhn/blog/2014/06/09/do-not-need-cla.html
[32] The is an argument that they can still include implicit patent grants – BSD, MIT – see http://lu.is/blog/2016/10/31/reacts-license-necessary-and-open/ + https://www.elcaminolegal.com/single-post/2016/10/04/Facebook-Reactjs-License ("I've never heard any lawyer postulate that [the BSD license] does not grant a license to fully exploit the licensed software under all of the licensor's intellectual property. … Developers-licensees (or, more to the point, their lawyers) have traditionally been very confident that the BSD License does not leave room for a licensor to successfully sue under patents."); On problems with implied licenses – see http://www.ifosslr.org/ifosslr/article/view/107/205
[33] See https://github.com/facebook/react/blob/b8ba8c83f318b84e42933f6928f231dc0918f864/PATENTS "The license granted hereunder will terminate, automatically and without notice, if you (or any of your subsidiaries, corporate affiliates or agents) initiate directly or indirectly, or take a direct financial interest in, any Patent Assertion: (i) against Facebook or any of its subsidiaries or corporate affiliates, (ii) against any party if such Patent Assertion arises in whole or in part from any software, technology, product or service of Facebook or any of its subsidiaries or corporate affiliates, or (iii) against any party relating to the Software."
[34] See https://medium.com/@dwalsh.sdlr/react-facebook-and-the-revokable-patent-license-why-its-a-paper-25c40c50b562 ; https://medium.com/@raulk/if-youre-a-startup-you-should-not-use-react-reflecting-on-the-bsd-patents-license-b049d4a67dd2 ; http://lu.is/blog/2016/10/31/reacts-license-necessary-and-open/ ; https://medium.com/@shanecurcuru/three-reactions-to-the-facebook-patents-license-b64e6942012b

project under MIT license after seeing an increase in number of communities opting to not use the project's code.[35]

To contrast this, under Apache v2 which also includes a stronger patent retaliation clause, each contributor grants royalty-free licensing of its patents concerning the project. Moreover, its retaliation clause creates a defensive mechanism under which patent litigation instituted by any licensee against "any entity (..) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement", then "any patent licenses granted" to such work automatically terminate. This essentially assures that any third party patents against anyone over the project will be retaliated by termination of any inward patents licenses received. This basically acts as a trigger deactivating the entire royalty-free patent shield over the project against a particular firm. As a consequence, all the patent owners are free to enforce their patents against it.

In its effects, we witness reliance on a similar mechanism also as an add-on practice that accompanies open source licensing from the outside. Open Invention Network is the most famous defensive patent pool which first obtains patents of its members and then licenses them to its members and the outside world for free under its OIN License Agreement.[36] The license includes a retaliation clause which [XX]

+ community norms

+ common cure rights commitment https://www.redhat.com/en/about/press-releases/technology-industry-leaders-join-forces-increase-predictability-open-source-licensing

2.3 Comparison

In a way, standards and open source projects are incomparable. While standards provide basic specifications which can support products commercialized via multitude of business models, open source *is* a set of pre-defined business models. The idea is of standardization is to cooperate on basic building blocks, and compete on their product implementation. Therefore while standards leave the choice of appropriation strategies wide open, open source licensing of implementations *is* one such strategy. While open source licenses that cover patents and copyrights significantly limit licensing-based business models, copyright-only licenses leave patent-licensing intact. That being said, the view of standardization as business model agnostic can be challenged if the SDO's IPR policy is more prescriptive. For instance, W3C's IPR policy requires royalty-free patent licensing.[37] The outcome is that appropriation strategies for standard-essential patents shrink, but are still left open for any non-essential specifications and product-related elements. In the conclusion then, open source is more limiting for the appropriation strategies not only because of its default IPR position, but also due to the fact that it covers also non-essential specifications and product elements.

At the same time, open source and standards share some features. According to Douglas Heintzmann from IBM, "[t]he similarity [of open source] to open standards lies in the development-by-community

[35] See https://code.facebook.com/posts/300798627056246/relicensing-react-jest-flow-and-immutable-js/
[36] See http://www.openinventionnetwork.com/joining-oin/oin-license-agreement/
[37] REF: W3C

approach, which makes the process visible to all interested parties."[38] The open way of production is, however, far from identical. Standards are adopted by SDOs after deliberations of participants in technical committees who reach a *consensus*. Open source projects, on the other hand, are administered by their maintainers. Before a new contribution is accepted, there might be an earlier discussion on the project mailing list.[39] Some projects also pre-announce in their guidelines how a contribution is reviewed and what types of contributions are accepted.[40] Although consensus is thus built differently, if community cannot agree on some contributions or their general trajectory, some members might decide to split the project and continue to develop it separately (so called 'fork'). This is different from standardization, where lack of consensus leads to no standard being produced. At most, participants can change the forum to a different SDO.[41]

This leads to another important distinction. Versions of standards are more stable than releases of open source projects. As explained by Lundel, "releases of OSS occur at more or less arbitrary points in time. Standards, on the other hand, are by and large static, or perhaps semi-dynamic".[42] Standards are valued exactly because they codify industry knowledge and provide firm reference points. Open source is more volatile in this regard. As will be shown, however, this aspect can drive synergies between two ecosystems by providing means of pre-testing or show-casing standards. Egyedi points out that many SDOs neglect implementation issues and concludes that '[w]here interoperability is concerned, standard development and implementation issues cannot be meaningfully separated'.[43] She therefore calls on SDOs in the area of ICT 'to shift their emphasis from standard development to a more systematic inclusion of implementation concerns, both at the technical level of standard committees and at the policy level of standard organizations'.[44] This seems to support this proposition as well when he argues that a new transparent way how to improve standardization process.[45]

According to Lundel, for instance the practice of tracking is a governance feature from open source world that was then adopted by SDOs such as W3C, IETF, and OASIS.[46]

3. Interactions

Previous discussion organically leads to natural question how do open source projects and standards interact in practice. According to Lundell and Gamelielsson, we can broadly distinguish the following three situations: (1) "standard first", (2) "software implementation first" and (3) "standard and implementation

[38] ETSI 2005 Report, p. 26
http://www.etsi.org/website/document/workshop/sosinterop/sosinteropiiibackground01.pdf
[39] See https://opensource.guide/best-practices/#what-does-it-mean-to-be-a-maintainer
[40] See for instance https://jekyllrb.com/docs/contributing/
[41] REF: to examples
[42] [30]." REF TO: Jakobs, K. 2006. ICT Standards Research -Quo Vadis? Homo Oeconomicus 23, 1, 79-107.
[43] Egyedi, T. (2007). Standard-compliant, but incompatible?! Computer Standards & Interfaces, 29(6), 605–613. doi:10.1016/j.csi.2007.04.001
[44] Ibid.
[45] REF
[46] Lundel,

of standard in parallel".[47] Daniel Veillard from RedHat offers similar categorization calling it: (1) "late implementor", (2) "early implementor" and (3) "software and stand in parallel".[48] In the following section, we will use this established classification to discuss some basic interactions.

.

3.1 From specification to implementation

The most typical scenario imagined by many when thinking about interaction between open source projects and standardization is when open source constitutes means of standard implementation. Open source implementation then competes with other implementations on the market. For instance, famous IEEE 802.11 standard is a set of specifications defining wireless local area network computer communication in specific frequencies. It is naturally also implemented in open source-based operating systems, including those using Linux kernel.[49] While typically such implementations are produced independently by firms active on the market or communities, increasingly, SDOs themselves are interested in designing their own reference implementations along with standard specifications.

This has several reasons. First of all, SDOs developing specifications connect participants that possess lot of knowledge in the area so it is natural to offer them a platform to further development of the technology.[50] Second, specifications are often too abstract and thus there is no guarantee of seamless implementation without an authoritative reference implementation.[51] A great example of this are problems encountered by two open source projects when trying to implement ISO's PDF standard.[52] West explains this as follows: [53]

> "for complex digital systems standards, the formal specification is inherently incomplete and the actual standard is defined both through the written specification and through actual implementations... for any firm trying to implement a standard, knowledge of both the formal specification and existing implementations is valuable. Otherwise, the implementer faces an extended trial-and-error process as it seeks to discover how other firms have resolved specification ambiguities. So a typology of openness must consider the openness both of the specification and implementation."

For these reasons, for instance, IETF requires multiple interoperating implementations before adopting any standard.[54] Third, SDOs which are able to offer reference implementation are more likely to see

---

[47] B. Lundell and J. Gamalielsson, "On the potential for improved standardisation through use of open source work practices in different standardisation organisations: How can open source-projects contribute to development of IT-standards?," in EURAS Proceedings, 2017, pp.137–155.

[48] Partly inspired by Redhat presentation > http://veillard.com/Talks/CLKBeijing2012.pdf

[49] See https://rtime.felk.cvut.cz/publications/public/ieee80211p_linux_2014_final_report.pdf

[50] See JRC study;

[51] Egedy?? He incompatible paper;

[52] Jonas Gamalielsson and Bjorn Lundell, Experiences from implementing PDF in open source: Challenges and opportunities for standardisation processes http://ieeexplore.ieee.org/document/6774572/?part=1

[53] West, Joel. 2004. "What are Open Standards? Implications for Adoption, Competition and Policy". Standards and Public Policy conference, Federal Reserve Bank of Chicago. May 11: Chicago, Illinois. Available online (last accessed on 24th

[54] See https://tools.ietf.org/html/rfc2026

successful market diffusion of their standards since such references test specifications and save investments in the implementation phase.

An excellent example of this such practice is MANO Open Source (OSM) project hosted by ETSI. In 2016, ETSI director created an Open Source Group OSM, the goal of which was to create open source reference implementations for ETSI's specifications in the area of network virtualization known as ISG NFV architectural framework.[55] According to its terms of reference, which also include number of governance agreements:

> "OSG OSM will provide an opportunity to capitalise on the synergy between standardization and open source approaches by accessing a greater and more diverse set of contributors and developers than would normally be possible. This maximises innovation, efficiency and time to market and ensures a continuing series of true (conformant) reference implementations."

OSM project is licensed under Apache v2, a permissive OSI-certified license, with royalty-free licensing of patents and stronger patent retaliation clause.

[previous developers, etc., how it improves; bit form the web + license;]

[main issue here, prevent non-compatibility, reducing uncertainty through pools/commitments?,]

3.2 From implementation to specification

Second, perhaps less frequent scenario is when open source project develops specifications which are then codified through specifications in the actual standard. An example is offered by JavaScript, which is one of the core World Wide Web technologies, along with HTML and CSS. JavaScript was first released by Netscape and SUN in 1995. Already in the original announcement, the two companies wrote:[56]

> "Netscape and Sun plan to propose JavaScript to the W3 Consortium (W3C) and the Internet Engineering Task Force (IETF) as an open Internet scripting language standard. JavaScript will be an open, freely licensed proposed standard available to the entire Internet community. Existing Sun Java licensees will receive a license to JavaScript. In addition, Sun and Netscape intend to make a source code reference implementation of JavaScript available for royalty-free licensing, further encouraging its adoption as a standard in a wide variety of products."

Eventually, unlike HTML and CSS that were standardized with W3C, the standardization work for JavaScript was commenced at ECMA International under the name of ECMAScript.[57] ECMAScript is then released under the BSD license applying to all the software,[58] however, with an explicit patent carve-out.

---

[55] https://portal.etsi.org/Portals/0/TBpages/OSM/Docs/ETSI_OSG_OSM_ToR_2016-02-09.pdf

[56] See https://web.archive.org/web/20070916144913/http://wp.netscape.com/newsref/pr/newsrelease67.html

[57] There is an interesting trademark history behind this naming choice, see https://www.infoworld.com/article/2653798/application-development/javascript-creator-ponders-past--future.html

[58] http://www.ecma-international.org/ecma-262/5.1/ ("All Software contained in this document ("Software)" is protected by copyright and is being made available under the "BSD License", included below. This Software may be subject to third party rights (rights from parties other than Ecma International), including patent rights, and no licenses under such third party rights are granted under this license even if the third party concerned is a member of Ecma International.")

For patents essential to the standard which are contributed by 'any party participating in a technical committee', regular ECMA patent policy applies.[59] Since ECMA Code of Conduct in Patent Matters allows for royalty-free or FRAND commitment, contributors are free to choose.[60] The standard is nowadays implemented by number of firms, such as Adobe, Apple, Google, Microsoft and the Mozilla Foundation. Each of the firms then determines its own license.

Another example of this scenario is standardization project called Linux Standard Base (LSB). It is a joint effort of several open source projects hosted by the Linux Foundation. Its goal is to develop and promote set of specifications that will increase compatibility among different operating systems (distributions) based on Linux. The results of the standardization work are published as ISO/IEC 23360-1:2006 as a standard document.[61] IEC website does not list any related patent declarations.[62]

As the two examples show, this scenario seems to mostly arise in situations when existing product serves as a starting point for a new standardization work or when family of similar products seeks to find a common building blocks in order to increase their mutual interoperability. Depending on timing of contribution and of the feedback loops, it can overlap with the third scenario.

[main issue here, copyright on specifications, but also IPR policy v open source license; since baseline, not big issue]

3.3 Co-development of specification and implementation

While first and second scenario appear to be almost linear, the third scenario certainly isn't. It covers situations when work on standardization and implementation are undertaken in parallel. The typical case is when during the standardization work, an early open source reference implementation work is developed, in order to see the functioning of the specifications in action. Gamalielsson and Lundell (2013)[63] covers an interesting case study of this regard. They study example of mutual interactions between an open source community, Drupal, and standardization community at W3C concerning RDFa technology, which is a set of extensions necessary for embedding rich metadata within web documents. Through study of Drupal and W3C trackers for RDFa, they find clear evidence of reciprocal actions between two communities.[64] Gamalielsson and Lundell summarize this as follows: 'standards can influence implementations of standards, implementations of standards can influence standards, and that implementations of standards can influence other implementations of standards'.[65] Moreover, according

[59] http://www.ecma-international.org/memento/codeofconduct.htm

[60] According to the registry, for instance, Intel and salesforce.com decided to grant royalty-free licenses – see http://www.ecma-international.org/publications/files/ECMA-ST/Ecma%20PATENT/EcmaListofPatentStatements.htm

[61] See https://www.iso.org/standard/43781.html ("International Standard ISO/IEC 23360-1 was prepared by the Free Standards Group and was adopted, under the PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, Information technology, Subcommittee SC 22, Programming languages, their environments and system software interfaces.")

[62] http://patents.iec.ch/iec/pa.nsf/pa_h.xsp?v=0#

[63] REF Lundel 2014

[64] REF

[65] J. Gamalielsson, B. Lundell, Experiences from implementing PDF in open source: challenges and opportunities for standardisation processes, in: K. Jakobs, (Ed.), Proceedings of the 8th IEEE Conference on Standardization and Innovation in Information Technology (SIIT 2013), ISBN 3-86130-802-9, IEEE, Piscataway, 2013, pp. 39–49.

to their findings, this set-up seem to have attracted new pool of participants to the standardization project at W3C. In a way, even two earlier scenarios can lead to co-development in the long run. Even already mentioned Mano project by ETSI, recognizes that 'essential feedback' to standardization work is one of its goals.

[sharing of contributions, parallel consistency of rules,]

4. Frictions Among IPR Policies

When standardization and open source projects interact, frictions between their accompanying policies can arise. This part explores what types of frictions do we see, and what are their consequences.

4.1 Scenario by Scenario

To begin with, let's have a look at the 'late implementor' scenario. Since implementation comes only after specifications are adopted in standardization process, the main question is whether open source licensing can be applied to specifications including SEPs with FRAND commitment. This is usually presented as the main point of potential friction. It is argued that while standardization community primary is after *fair* access, open source community usually mandates *free* access. In fact, this friction has two main components: (a) legal compatibility and (b) business consistency. As will be shown, with many open source projects, the business inconsistency is the key issue.

In the late implementor scenario, patent holders holders usually do not need to worry. After all, competitors will have to license any of their used patents anyway, regardless whether such project is open source or not. Actually, the open source character can improve chances of detection of an infringement. Their concerns start when the project is being developed as a reference implemention by SDO itself. In these cases, they face a choice of whether to participate or not. If they do, depending on the project's license, they might need to license their SEPs, or even other non-essential patents reading on the reference implementation. Moreover, they might expose themselves to patent retaliation clauses which, again depending on the wording of the license, could be used as a leverage even in other cases, where they may seek royalties.

In the early implementor scenario, the project license can influence the viability of the follow-on standardization in two ways. If a standardization project is initiated, and the project license did not include any grants concerning patents reading on the implementation, the standardization process will need to take care of the all patents that remain essential to standard-specifications. The exact way would depend on particular SDO and its policy, of course. However, the lack of participation by original developers in the follow-on standardization could endanger its viability as they are neither obliged to consent to inclusion of their patents to a standard, nor to provide a FRAND commitment. In a way, they could prevent successful open source projects from being successfully standardized into building blocks. Naturally, if the original open source project license includes a royalty-free patent grant, the follow-on standardization is easy to arrange to the extent that newly minted SEPs are already covered by it. In these cases, the project license basically serves as a substitute for such standardization-specific commitment.

As regards copyrights, if code from the original project becomes part of the specifications, two copyright policies need to be legally, but also business compatible. For SDOs that earn their living by selling standardization documents, the decision to include open source code could mean that particular standards, or their parts (depending on the open source license and scope of use), have to be available

free of charge. They might be thus legally constrained in introducing pay-walls for their standardization documents. In addition, their standard contributor copyright policies in the technical committees need to be compatible with the original project license.

In the co-development scenario, the interactions of the first two scenarios are only accelerated. The flow of exchange between specifications and implementations goes both ways and at much higher speed. Consistency of two policies and legal certainty is then highly important for successful standardization efforts and clear understanding of the rules between participants of two communities. Naturally, if the participants in both open source project and standardization work are identical, all problems are mitigated. By when such membership is not identical, two ecosystems might contain stakeholders with both overlapping and antagonistic interests. For instance, open source developers might be interested to assure absolutely royalty-free diffusion of implementations, while standardization contributors want to capitalize on their investment embodied in the included technologies. Although both parties desire diffusion, they do so under different conditions, and with different business models in mind.

4.2 Legal incompatibility

In the last couple of years, it was often debated whether open source is compatible with FRAND. Much of the debate seem misleading because authors disagree on when two systems are incompatible. Kesan, for one, for instance says that: '[o]pen source licenses cannot be categorically described as conflicting with a FRAND-based standards license unless each term of each open source license is examined and tested against the FRAND model.'[66] In other words, according to his definition, as long as there is a single open source license out there, in his case certified by OSI, that is compatible with an unspecified type of FRAND commitment, one cannot say that two systems are incompatible. Some broaden this further by suggesting that there are moreover many open source licenses that are not OSI-certified and would be compatible.[67] Others argue that the mere fact that the patent holders are not able to collect royalties means that the two are incompatible.[68] The result of this is that the while one camp claims that 'an empirical examination of all OSS licenses approved by the OSI reveals that there is no inherent conflict between OSS and FRAND',[69] the other camp disagrees by saying that FRAND is incompatible most of the time.[70] Who is right?

Firstly, the problem is definitional. Two licenses are *legally incompatible* only if their obligations cannot be reconciled at the same time. For instance, when once license requires disclosure of a component and another prohibits it, it is clear that there is an incompatibility. However, if two licenses can be respected at the same time, by settling for the least common denominator, then there is no legal incompatibility. So for instance, if a commitment to be fair is rivaled by a commitment to not charge anything, there is no conflict between the two as one can simply not charge anything and thus remain also fair. In these situations, one may speak of (business) inconsistency, or frictions, but hardly about legal incompatibility. In the debates, this distinction often gets lost.

Second, the problem stems from the broader framing. As was explained earlier, there are hundreds of open source licenses and dozens of FRAND commitments. One can answer any legal compatibility

[66] KESAN
[67] REF:
[68] REF
[69] KEsan
[70] See https://fsfe.org/news/2016/news-20160428-02.en.html + https://fsfe.org/activities/os/why-frand-is-bad-for-free-software.en.html#fn1

question only with reference to two specific legal texts. This also means that answering the question about compatibility between open source and FRAND in general is a pointless exercise. No other answer can be given than the lawyerly 'it depends'. What is reasonably certain is that we encounter both situations: legally compatible and incompatible licenses with various FRAND commitments.

Third, the problem also seems interpretational. For at least some types of conflicts, reasonable minds might differ. Even thought people agree that GPL family of licenses is incompatible due to liberty and death clause, it is less whether

- Only some provisions, GPL family
- To some extent patent retaliation clauses?
- OSI covers patents or not? [Marr, no; respondents JRC, yes]
- Defensive Patent pools on top? Vs retaliation clauses?
- IoT case see

4.3 Business inconsistencies

Standardization allows firms to cooperate on specifications and compete on their implementations. Where firms aren't vertically integrated on the product markets, they can simply earn by licensing their patents.[71] Naturally, this strategy is heavily constrained if the SDO's IPR policies require them to license standard-essential patents on royalty-free basis. For 'pure innovators',[72] i.e. non-integrated up-stream firms, such licensing arrangement can significantly reduce their appropriation strategies. Open source licensing that includes royalty-free patent clauses leads to comparable business outcomes, although the scope and therefore impact of thereby licensed patents differs. The opposition to patent royalty-free SDO policies thus logically also extends to patent royalty-free open source licenses. The argument against such licenses is broader because open source licensing might limit appropriation strategy for technology and products. Although firms might still 'implement-around' the reference implementation, it might be hard, as an early royalty-free competing product, to beat on the market.

- Business model neutral
- Patent retaliation clauses & enforcement strategy
- Copyright and sales policies of traditional SDOs

4.4 Innovation opportunities

From the above analysis, it is clear that SDO choices have consequences for the innovation ecosystem. They incentivize or disincentives different sets of innovators.

The emphasis on royalty-free patent licensing is likely to discourage 'pure innovators' from participating in either of two ecosystems.[73] Following W3C's 2002 change to royalty-free policy, it was reported in the

[71] About advantages and disadvantages of vertical integration see https://pure.uvt.nl/ws/files/992802/2008-018.pdf

[72] Ibid

[73] Kapos ("moving to incompatible licenses for FRAND standards submissions weakens innovation incentives and discourages innovators from participating in standardization efforts")

media that large patent holders such as IBM, SAP, and Microsoft decided to channel some of their standardization work concerning Web technology to OASIS, which had more favorable policy at the time. The real evidence for this type of discouragement can be found in the studies which look at ecosystem changes after SDO's revise their IPR policies in this way. Very illustrative is study conducted by Stoll who empirically examines effects of IPR policy change by OASIS in 2005.[74] He finds that the change from FRAND to royalty-free (RF) licensing regime is correlated with a significant decrease in the overall membership. However, at the same time, he finds that the composition re-shapes following the change as the share of non-profit research organizations and systems integrators significantly increases in the aftermath. Polhmann (2017)[75] and Contreras (2013)[76] offer two similar empirical studies without finding similar effects. However, both of them focus on SDO changes at IEEE and VITA concerning what constitutes fair, reasonable and nondiscriminatory licensing rates.

Royalty-free patent policies may discourage 'pure innovators'. It might potentially limit technologies of new generations brought to SDOs by these firms since it restrains their appropriation strategies. One could even say that it pushes 'pure innovators' outside of the standardization system and incentivizes them to either vertically integrate or negotiate with such integrated firms. Inclusion of royalty-free patent licensing on implementations only further deepens this effect. At the same time, royalty-free policies also accelerate diffusion and facilitate more follow-on innovation.[77] This is especially so if mere royalty-free specifications are coupled with royalty-free reference implementations. Implementations provide a valuable way of testing standards and of providing building blocks which further save resources for follow-on users and innovators. Which effect is stronger is hard to tell. It is also hard to tell whether the new types of innovators attracted to royalty-free system can off-set the type of contributions previously provided by original 'pure innovators'.

Moreover, it should be emphasized that SDO choice, if accompanied by particular open source licenses, can active a new pool of innovators, i.e. software developers. As shown by Belenzon and Schankerman, open source 'developers strongly sort by license type, project size and corporate sponsorship' due to heterogeneity of their motivations.[78] In their empirical study they find that highly restrictively licensed projects (e.g. under GPL) attract almost exclusively developers who contribute only to projects with such restrictive licensing. On the other hand, contributors to projects with permissive licenses (e.g. Apache, BSD, MIT) are more likely to contribute to larger projects and to those that are sponsored by corporations. This reveals split in the underlying motivation of open source developers. Highly restrictive (strong copyleft) licenses seems to attract motivated agents who are dedicated to the movement's ideology, while permissive licenses attract developers who want to improve their career prospects by building up own reputation. Moreover, Belenzon and Schankerman find that developers contributing to restrictive projects

[74] Stoll, Thimo Pascal, Are You Still in? – The Impact of Licensing Requirements on the Composition of Standards Setting Organizations (December 8, 2014). Max Planck Institute for Innovation & Competition Research Paper No. 14-18. Available at SSRN: https://ssrn.com/abstract=2535735 or http://dx.doi.org/10.2139/ssrn.2535735

[75] Pohlmann, T. 2017. "Empirical Study on Patenting and Standardization Activities at IEEE," https://asoft20107.accrisoft.com/atfrand/clientuploads/news/IPlytics_2017_Patenting%20and%20standardization%20activities%20at%20IEEE.pdf

[76] Contreras, J., "Technical Standards and Ex Ante Disclosure: Results and Analysis of an Empirical Study", Jurimetrics, Vol. 53(1)

[77] Jorge, W3C and IETF paper, p. 881

[78] Belenzon, Sharon and Schankerman, Mark A., Motivation and Sorting in Open Source Software Innovation (October 2008). LSE STICERD Research Paper No. EI47. Available at SSRN: https://ssrn.com/abstract=1401776

are more likely to be involved in projects close to end consumer, while developers contributing to unrestrictive projects are more likely to be involved in developer-oriented projects. In other words, their findings suggest that more fundamental research R&D takes place within permissively licensed projects whose developers are motivated by the careers prospects within industry. This an important finding for standardization as it would suggest that basic infrastructure projects that aren't user-oriented are anyway less likely to attract developers with preferences for strong copyleft licenses. If this conclusion is correct, then opportunity costs of SDO's choice of permissive open source licenses would be very low. It remains open to what extent permissive open source licenses when adopted by SDOs for their reference implementations attract also firms that previously weren't involved in the standardization work.

Although we lack comparable studies for self-selection of individual developers or firms depending on the underlying patent policy of projects, there is some anecdotal evidence suggesting that even broad retaliation clauses might discourage some innovators from adopting or further developing technologies. In 2017, Facebook faced this problem with respect to a broad patent retaliation clause in its React project license which was permissive. This customized clause would automatically terminate authorization to use any of the Facebook's React-relevant royalty-free licensed patents if any potential user wants to enforce any of its own patents against Facebook. The company faced strong backlash from the community and eventually decided to re-license its project because number of open source projects announced that they will not use the technology or build upon it.[79]

CHOICE >>>

a) User-oriented, copyleft
b) Infrastructure, but substitute, permissive
c) Infrastructure, but non-substitute in software world, then permissive, without

5. Conclusions

[79] https://code.facebook.com/posts/300798627056246/relicensing-react-jest-flow-and-immutable-js/