Defining Open Standards

Lawrence Rosen
Rosenlaw & Einschlag

ABSTRACT: *There is no agreed upon definition of open standards, and so we don't always mean the same thing when we talk about them. In this article, Lawrence Rosen proposes a set of principles for open standards, and explains how a clear definition will encourage the development of open standards that can be implemented in both open source and proprietary software.*

> **def·i·ni·tion** (*noun*)
> A statement expressing the essential nature of something.[1]

A definition gives us a means to distinguish those things that have something in common from those that do not. Without definitions—or with different definitions—our conversations are adrift. It is not possible to speak of a subject without being able to define it, or we may all be speaking of entirely different things.

That is why I am so concerned that there is no agreed upon definition of open standards. Because of this, the conversations we have in the standards world about the results of industry standards setting are often set-speeches of people talking past each other rather than dialogue about how to achieve open standards.

The problem is not that standards organizations can't define important concepts when they set their best minds to it. Here, for example, is a definition of "Essential Claims" from the W3C Patent Policy:

> Essential Claims shall mean all claims in any patent or patent application in any jurisdiction in the world that would necessarily be infringed by implementation of the Recommendation. A claim is necessarily infringed hereunder only when it is not possible to avoid infringing it because there is no non-infringing alternative for implementing the normative portions of the Recommendation. Existence of a non-infringing alternative shall be judged based on the state of the art at the time the specification becomes a Recommendation.[2]

This demonstrates that standards professionals can write precise legal definitions. Yet despite our obvious ability to create definitions when appropriate and to write policies around them, standards organizations often refuse to define "open standards." This seems to be a policy problem rather than a skills problem.

It is certainly a policy problem for the community of open source companies and software professionals around the world when standards organizations refuse to define open standards. We then have no easy way to distinguish standards we can implement under our software licenses and our development methodologies from those we cannot.

It is no longer enough to assert, as was said recently on one standards IPR discussion list, that "[our standards organization] has created open standards since 17 May 1865."[3] Technology has evolved significantly since 1865, and what was open at the time of steam locomotives and the American Civil War may not be considered open today. Software is fundamentally different from the technologies that came

before it. For one thing, it is both copyrightable and patentable and, with appropriate locks in some jurisdictions, can even be kept a trade secret after it is sold. Software standards are fundamentally different from standards for electrical outlets, including perhaps in their necessary degree of openness.

Which of the more than one million supported standards in the world today[4] are open and which are not? Wikipedia, the collective wisdom of anonymous self-appointed subject experts, defines open standards in the least useful way as "publicly available specifications for achieving a specific task."[5] By this definition, every one of those million-plus specifications published by standards organizations would be open. How useful is a definition that includes all standards and excludes none?

Some argue that open standards are all about process. ANSI, for example, understands the term "to describe a collaborative, balanced and consensus-based approval process for the promulgation of domestic or international standards."[6] While process is obviously important—we use the word "open" to indicate things that are not done in a hidden, secret, or private way—process alone does not necessarily an open standard make.7 The semi-public processes for creating standards do not by themselves guarantee that all can implement them without paying onerous patent royalties or without imposing impossible business conditions or burdens on some implementers. Are standards open if they are burdensome? Which burdens are acceptable, which are not?

This becomes particularly important when we deal with software standards that are to be publicly available for implementation in open source software. Software, and in particular open source software, is now a ubiquitous and essential component of all technology. If you cannot implement a software standard in open source software because of the license terms under which the standard is available, in what sense can the standard still be called open?

Freedom to Implement

Some have argued that not all software needs to be implemented under open source rules and therefore the linkage between open source and open standards is unnecessary or misleading. If so, which software standards should be off-limits for open source, and who decides that? Should industry standards organizations or the large corporations that often run them make the decision for everyone which software can be open source? Obviously not. Any meaningful definition of open standards should recognize that open source and proprietary software compete. Preferences for one or the other in the standards arena are intolerable.

The simplest responsible way to resolve the linkage between open source and open standards is by a definition that asserts no preference for open source over proprietary implementations. I suggest the following basic principle: ***Everyone should be free to implement open standards in both proprietary and open source software, or the standards should not be called open.***

The phrase "free to implement" is, of course, the sticking point in that sentence. It must be understood in the context of the definition of open source software, which has its own ideas of software freedom. But we must not forget proprietary software. Nothing I suggest below regarding a definition of open standards will disadvantage proprietary software in any way.

Open Source

We can be precise about what the term "open source" means, because there is a definition, established by consensus, published for all to see[8], and used as the basis for distinguishing open source from other software. In summary, these are the principles[9] that apply to all open source software:

1. Licensees are free to use open source software for any purpose whatsoever.
2. Licensees are free to make copies of open source software and to distribute them without payment of royalties to a licensor.
3. Licensees are free to create derivative works of open source software and to distribute them without payment of royalties to a licensor.
4. Licensees are free to access and use the source code of open source software.
5. Licensees are free to combine open source and other software.

In order to create an open standards regime that is appropriate for open source implementations, we must ensure that licenses needed to implement the specifications for the standards—licenses for necessary copyrights and for essential claims of patents—are available under terms and conditions compatible with open source licenses. And of course, to avoid preferences for open source, we must also ensure the same copyrights and patent claims needed for the standards are available under acceptable terms for proprietary implementations as well.

Royalties and Open Standards

The open source principles listed above have important implications for open standards. In particular, licensees must be free to copy, modify, and distribute open source software "without payment of royalties to a licensor." No revenue from royalties is available in those channels of commerce to pay fees to practice the standard, and for most such software the distribution is free. This fundamental fact of open source is incompatible with royalty-bearing standards.

On the other hand, the major paradigm in the standards world has long been "reasonable and non-discriminatory" (RAND) royalties. But non-zero RAND royalties simply don't work for open source. That is why open source organizations have consistently opposed the RAND policies of certain software standards organizations.[10] To call a RAND standard "open" would be to violate the principle I proposed earlier, namely that an open standard should be equally available for open source and proprietary software.

RAND is only compatible with open source if the "reasonable" royalty is zero. Some have suggested the compromise term RAND-Z so as not to discomfort those who still love the term RAND, but that hasn't caught on. I see little reason to be subtle on this point and so I use the term "royalty-free" consistently to make it clear that royalties are incompatible with open source software licensing models. Royalty-free is a difficult requirement. Some patent and copyright owners simply refuse to allow their intellectual property to be practiced royalty-free for implementations of industry standards. This is indeed a problem, but many of us are resigned to living with it as long as we have software patents. Forcing patent owners to donate their patents for industry standards will require legislation, an unlikely prospect.

Meanwhile, we need a clear definition that allows us to distinguish royalty-bearing standards from the others. If you want, call standards that bear royalties "RAND standards." ***Open standards should be available to everyone on royalty-free terms, or the standards should not be called open.*** That is one way a clear definition can help distinguish among standards.

Open Standards Guarantee Freedom to Participate

Standards experts are quite right to point out the importance of the process for developing and promulgating the standards. Without an open process, there is no way to judge the trustworthiness or utility of the standard.

This is entirely consistent with the open development models that guide the work of most open source projects. All successful open source software, like all successful software standards, are developed using collaborative, balanced, and consensus-based processes. There is wide support in the software community, among both open source and proprietary participants, for open standards processes.

I therefore urge that the definition of open standards include this principle: ***Open standards should be developed using a collaborative, balanced, and consensus-based approval process, or the standards should not be called open.***

This is not, of course, merely a surface characteristic of a standards organization. Openness must permeate the organization at all levels, from the time the standard is first discussed to the time the standard is ultimately adopted. Openness of process ensures openness of result. For example, the open discussions and debate in JEDEC around the setting of a standard for computer memory, at the same time that Rambus was allowed to be secretive about intellectual property it contributed to the standard, led to the debacle of the *Rambus* cases. Open processes cannot tolerate certain kinds of secrecy. The processes for members' disclosure of intellectual property claims on standards, and any mandatory licensing rules that would apply in the event they fail (or refuse) to disclose, are as important aspects of an open process as is the method of voting on the technology alternatives.

Open processes should also apply when a standards organization negotiates licenses for the intellectual property of third parties necessary to implement the standard. Negotiations in secret may lock out some potential implementers and users of the standard. Now that this form of openness has been generally recognized as acceptable,[11] standards organizations should formalize their procedures for doing so.

That is why I suggest that an open standards process should also include the principle: ***Open standards should be developed under formal and binding commitments for the disclosure and licensing of copyrights and patent claims, or the standards should not be called open.***

This way, process pays more than lip service to the openness goal, and the resulting standards will be freely available for implementation.

Patent Reciprocity and Open Standards

It is not enough that a standard be born open. It must also remain open despite the efforts of patent owners to charge royalties or to impose conditions on its implementation that disadvantage either open source or proprietary implementers. Otherwise, software that implements the standard may suddenly become unavailable or more expensive, with potentially large financial and business effects on the implementers and distributors of that software.[12] Remember, too, that although software vendors first pay the costs of non-open standards, it is ultimately the buyers and users of software who have the greatest financial and business incentives to keep them open.

This is a problem for both open source and proprietary vendors. We are all faced with the prospect of paying substantial royalties to patent holders for implementing standards we thought were going to be

royalty-free. Everyone needs defensive strategies to protect open standards from patent infringement claims. A joint defensive strategy would be particularly helpful. All who contribute to an open standard, all who implement it, and all who practice it, must act jointly to keep that standard open.

One common strategy in open source is a condition of *reciprocity*. Reciprocity is defined as "a mutual or cooperative interchange of favors or privileges."[13] In open source practice, it means that the same license under which the software is first distributed must be used in all subsequent downstream licensing transactions involving copies or derivative works. The GPL and several other major open source licenses[14]—which apply to most free software—impose reciprocity conditions upon downstream licensees who distribute copies or derivative works to third parties. Once the software is distributed for free, copies of the software itself and any derivative works of that software will remain free. In effect, reciprocity creates a common pool of copyrighted works and patent claims shared among all contributors and users.

So, too, reciprocity in open standards can have the effect of protecting the standard and creating a common pool of patent claims that are available for all who share the standard. There is no particular reason why that can't become a condition for participation in the standards process and a condition for commercial distribution of software that implements the specification.[15]

Reciprocity in open source takes diverse forms, and open source licenses differ in the scope and effect of their reciprocity provisions. So, too, are there differences among reciprocity provisions in patent licenses for standards. Regardless of the details, though, there is no room for provisions in licenses for open standards that place one party in a preferential position to another. Reciprocity in open source licenses is judged by whether it discriminates among potential licensees. Reciprocity in open standards should be judged the same way.

Reciprocity can also be an effective deterrent to nuisance provisions in patent or copyright licenses. For example, reasonable patent owners won't tolerate licenses provisions that require formal execution of patent or copyright licenses, or that implementers place notices in executable versions of software, if that means they too will be burdened by similar requirements from other patent and copyright owners. Such license provisions add too much friction to the development and distribution process. Perhaps we should require that reciprocal licenses be "reasonable."

Many open source licenses also include defensive termination conditions that have the effect of obtaining reciprocal cross-licenses of patents without the burden of saying so explicitly. For example, one typical license terminates automatically "as of the date you commence an action, including a cross-claim or counterclaim, against Licensor or any licensee alleging that the Original Work infringes a patent."[16] A licensee must choose whether to forgo enforcement against certain software use of its own patents (and in effect license their use), or sue for patent infringement and stop its own use of that software. That isn't often an easy choice, particularly when the patented software is an essential business or product component. That difficult choice, however, has the beneficial effect of defending the software from infringement lawsuits.

In the standards world, defensive termination is often built into some covenants not to sue. Both Sun[17] and Microsoft[18] recently published such covenants for proposed standard XML document formats. In each case, the covenant does not apply to anyone who sues Sun or Microsoft for patent infringement relating to their proposed XML standard.

Reciprocity of patent licensing, whether explicit in license conditions or implicit through defensive termination provisions, is apparently a shared goal of many in the open source and proprietary software business and many in the standards world. Based on the regularity of demands for reciprocity, it appears to be essential to the companies that own patents and are only willing to license them for industry standards upon reciprocal conditions.

I suggest the following principle: ***Open standards should be made available under reasonable reciprocal licenses that require licensees to share under the same terms their own patent claims reading on the standard, or the standard should not be called open.***

The result will be a commons of necessary patents that is available to all who use the standard, to everyone's mutual benefit.

Source Code and Open Standards

In the software field, it is often difficult to distinguish the specification of a standard from its implementation. The code to accomplish something in a standard way is typically the very code that is published in the specification. In some cases, the source code *is* the specification.

Even when a specification is not literally the computer instructions to accomplish a task in a standard way, the specification is often the most appropriate documentation of that software. For the best open source software, the implementation and its documentation are well-integrated, and both are often made available in source code form.

The implication of that open source requirement for standards organizations is profound. They must adopt copyright policies that permit their specifications to be copied, modified, and distributed in both open source and proprietary software and documentation, without payment of royalties.

Some argue that the right to modify a specification is unnecessary and counter to the policies of standards organizations to encourage conformance to the standard. Encouraging conformance is one thing; requiring that everyone use a specific piece of code to implement conforming software is another. All software improves upon the past, and restrictions on improvements to copyrighted software are counter to the freedom for open source licensees to create derivative works. Standards organizations need to find other ways to encourage conformance—such as trademarks or certification marks—rather than locking all implementers into using unmodified code.

The open source principles listed earlier don't imply that standards organizations cannot charge a reasonable fee for copies of any specifications that they themselves distribute. Open source software is not always zero price, and neither must copies of specifications of open standards be distributed at zero price. The rules for open source only prohibit licensors—in this case standards organizations—from charging royalties for copies and derivative works that others make and distribute on their own. As long as the price is reasonable and the quality of the specifications is assured, nobody will bother to subvert this important revenue stream that helps support the standards organization.

That's the successful way open source copyright licenses work, and there's no reason open standards shouldn't work the same way.

This permits a simple principle: ***The specifications for open standards should be available to everyone on open source copyright license terms, or the standards should not be called open.***

Summary: Open Standards Principles

In summary, here are the highlighted sentences from this article, rephrased as a proposed set of Open Standards Principles.

1. *Everyone should be free to implement open standards in both proprietary and open source software.*
2. *Open standards should be available to everyone on royalty-free terms.*
3. *Open standards should be developed using a collaborative, balanced, and consensus-based approval process.*
4. *Open standards should be developed under formal and binding commitments for the disclosure and licensing of copyrights and patent claims.*
5. *Open standards should be made available under reasonable reciprocal licenses that require licensees to share under the same terms their own patent claims reading on the standard.*
6. *The specifications for open standards should be available to everyone on open source copyright license terms.*

Anything else should not be called an open standard.

Postscript: An Open Standard Definition

I admit that I didn't deliver a definition of open standards, and I've never seen a really complete one from anyone else. The Samuel Johnson of the standards world has yet to be born.[19]

Instead, all I have suggested are principles for what open standards ought to be. I used the word "should" instead of "shall," and anyone familiar with legal terms understands that's not how conformance to standards is achieved in the world.

An agreed definition of open standards will require much more effort and talented draftsmanship. For example, the three-sentence definition of "Essential Claims" that I copied on the first page of this article was written collectively by a group of lawyers and standards professionals representing software companies and major open source projects. It was the result of careful negotiation over several months.

It will require an equally dedicated effort to achieve an acceptable definition of "open standards." I hope that the standards community can work together—using a collaborative, balanced, and consensus-based process—to define, in the best detail and clarity that we can achieve, the essential nature of open standards.

This article was written for publication in The Standards Edge series, published by The Bolin Group. Portions of this article were included in a presentation to the "Standardization: Unified or Divider" conference in Vancouver, BC, December 5-7, 2005. Copies of the slides for that presentation, also licensed under OSL 3.0, are available at ***http://www.rosenlaw.com/VANCOUVERpresentation.pdf.***

About the Author

Lawrence Rosen is both an attorney and a computer specialist. He is founding partner of Rosenlaw & Einschlag, a technology law firm that specializes in intellectual property protection, licensing, and business transactions for technology companies. In addition to this law practice, Larry also served for

many years as general counsel and secretary of the non-profit Open Source Initiative (OSI). He currently advises many open source companies and non-profit open source projects including the Apache Software Foundation. Larry serves on the board of International Characters and on the advisory boards of Spike Source, Black Duck Software and JasperSoft. He speaks around the world about open source and open standards. His book, *Open Source Licensing: Software Freedom and Intellectual Property Law*, was published by Prentice Hall in 2004. He is a lecturer in law at Stanford Law School.

---

[1] Merriam-Webster Online Dictionary, "definition of word: definition" **http://www.m-w.com/dictionary/definition**.

[2] World Wide Web Consortium, "W3C Patent Policy," **http://www.w3.org/Consortium/Patent-Policy-20040205/.**

[3] Private email. I do not disclose the source because this discussion took place on a private IPR discussion list at an international standards organization, and the email was disclosed to me in confidence. I took it as an example of how policies for "open" standards are often developed in secret by corporate professionals based upon antiquated opinions about the role of such standards in the modern world.

[4] Andrew Updegrove, "ConsortiumInfo.org, Standards Blog," **http://www.consortiuminfo.org/standardsblog/**

[5] Wikipedia contributors, "Open standard," *Wikipedia, The Free Encyclopedia,* **http://en.wikipedia.org/w/index.php?title=Open_standard&oldid=61198124** ["General disclaimer—Use Wikipedia at your own risk—Wikipedia does not give medical advice—Wikipedia does not give legal opinions—Wikipedia contains spoilers and content you may find objectionable. WIKIPEDIA MAKES NO GUARANTEE OF VALIDITY."] This definition of "open standard" was changed in Wikipedia since the time this article was written, leaving a still-moving target for a definition.

[6] American National Standards Institute (ANSI), "**Current Attempts to Change Established Definition of "Open" Standards," http://public.ansi.org/ansionline/Documents/Standards Activities/Critical Issues Papers/Open-Stds.pdf.**

[7] Even OASIS, a standards organization with a dedication to open processes, has a policy regarding the licensing requirements for its standards that in some cases precludes implementing those standards in open source software. OASIS, "Intellectual Property Rights (IPR) Policy," **http://www.oasis-open.org/who/intellectualproperty.php**. The battle that occurred upon the publication last year of their revised licensing policy demonstrated, if nothing else, that there was a strong difference of opinion in the software community about what the term "open standard" actually means.

[8] Open Source, "Open Source Definition," **http://www.opensource.org/docs/definition.php**.

[9] Lawrence Rosen, *Open Source Licensing: Software Freedom and Intellectual Property Law* (New York: Prentice Hall, 2004), Chapter 1.

[10] The reader is invited to search the Web for widespread public discussions of the W3C and OASIS patent policies and their implications for open source. For example, see Free Software Foundation, "A Call to Action in OASIS," **http://www.fsf.org/news/oasis.html**.

[11] Statement recognizing the procompetitive potential of royalty discussions in standards setting, under the antitrust "rule of reason," Deborah Platt Majoras, "Recognizing the Procompetitive Potential of Royalty Discussions in Standard Setting," (remarks at Standardization and the Law: Developing the

Golden Mean for Global Trade, Stanford University, Stanford, CA, September 22-23, 2005), **http://www.ftc.gov/speeches/majoras/050923stanford.pdf**.

[12] The *Rambus* and *Eolas* cases are recent examples where patent owners appeared late and caused great damage. Microsoft, too, may use its FAT patents to prevent standard ways of accessing its ubiquitous file system. Most users of software probably don't realize how integral industry standards are to their business and financial applications, nor how disruptive it might be to them if the standard suddenly became more expensive or less available.

[13] American Heritage Dictionary of the English Language, 4th edition, "definition of word: reciprocity," **http://www.bartleby.com/61/38/R0083800.html**.

[14] Open Source, "The Approved Licenses," **http://opensource.org/licenses/index.php**.

[15] Reciprocity is demanded in almost all patent grants made to the Internet Engineering Task Force (IETF) for that organization's standards. [*See* **https://datatracker.ietf.org/public/ipr_list.cgi**.] All of the major patent owners understand and routinely mandate reciprocity in their licenses.

[16] OSL 3.0 § 10, **http://www.rosenlaw.com/OSL3.0.htm**.

[17] OASIS, "Open Document Format for Office Applications (OpenDocument) TC," **http://www.oasis-open.org/committees/office/ipr.php**.

[18] Microsoft Office Online, "Microsoft Covenant Regarding Office 2003 XML Reference Schemas," **http://www.microsoft.com/office/xml/covenant.mspx**.

[19] Samuel Johnson, 1709-1784, was the author of the first dictionary of the English language that used quotations to illustrate usage. Published in 1755 with definitions for over 40,000 words, Johnson's Dictionary has served as the basis for all English dictionaries since. In English-speaking countries, the latter part of the eighteenth century is sometimes called the Age of Johnson in recognition of the importance of his defining work.