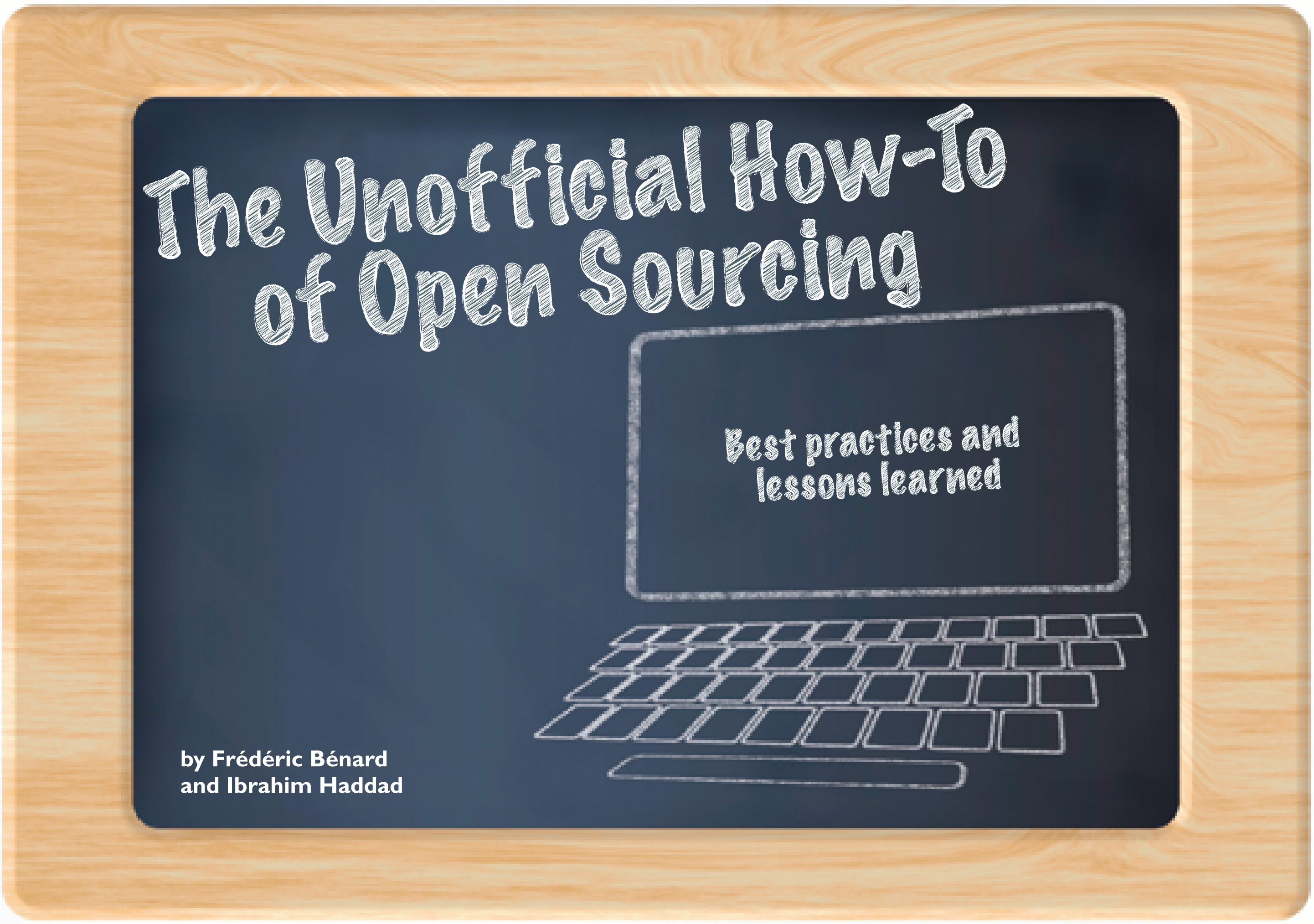

The Unofficial How-To of Open Sourcing

Best practices and lessons learned

by Frédéric Bénard
and Ibrahim Haddad

Open source software is shifting the software industry into a new paradigm, moving from developing proprietary code behind closed doors to developing code that can be shared, modified and redistributed openly. Key benefits associated with this shift is reducing development cost and software components complexity, developing re-usable common-off-the-shelf software assets, while increasing flexibility and using common enablers. Organizations that embrace the open source model and follow it when it influences positively their ways of building software, will increase their chances to retain their competitive advantage. In this article, we review some of the best practices to follow when taking a proprietary technology and making it open source.

1. Open Source for the Right Reasons

To be successful in open sourcing a project, you must have the right reasons or motivations. In a previous article published in Enterprise Open Source Magazine, we discussed the top good reasons to open source proprietary software, which included:

- Providing a reference implementation to a standard
- Ensuring that critical software remains viable
- Ensuring that new features are implemented
- Taking control of your own destiny
- Undercutting the competition
- Commoditizing a Market
- Partnering with others and promote goodwill for your company in the developer community
- Driving market demand by building an ecosystem

About the Authors

Dr. Frédéric Bénard is Engineering Manager at Motorola and leads the Open Source Software Center of Excellence, which is part of the Motorola "Embedded Systems, Open Source and Linux Technology Group". He holds a B.Sc. in Physics from McGill University, a M.Sc. and a Ph.D. in Physics from the University of Toronto, and an MBA from McGill University.

frederic.benard@motorola.com

Dr. Ibrahim Haddad is Director of Embedded & Open Source Technology at Motorola. Prior to Motorola, Dr. Haddad managed the Carrier Grade Linux and Mobile Linux Initiatives at the Open Source Development Lab (now the Linux Foundation). He received his Ph.D. in Computer Science from Concordia University in Montreal, Canada.

ibrahim.haddad@motorola.com

- Offering your customers the ability to support themselves and add their custom features

Open sourcing for the wrong reasons will not support your goal and will have a negative effect on your progress and relation with the open source community.

2. Legal Review and Understanding Intellectual Property Implications

The second step in the process is to audit the code base which you plan to open source thoroughly to verify that all of the code is owned by you or, in the case of open source packages that might be included in your code, that you have the appropriate distribution rights. This audit may be partially automated with the use of scanning tools that are commercially available today.

Furthermore, you must also evaluate if any of your intellectual property (IP) will be released as part of open sourcing the code. Note that, in the case of a large company, one division may not be aware of the IP from another division. In that situation, it is important to have your open source project reviewed by a group that is familiar with all of your company's IP.

3. Select an Open Source License

The adoption of a simple, well known and popular open source license will go a long way to encourage community participation in your project. Therefore, instead of creating your own open source license, it is most preferred that you use an existing license already approved by the Open Source Initiative (OSI). A list of OSI approved licenses can be found at http://www.opensource.org.

It is highly recommended to involve your company's legal department in the license selection process. In all cases, there are some general guidelines that we present below:

- Contribution to an existing open source project must follow the project's license.
- Contribution to the Linux kernel must be released under the GNU General Public License (GPL) Version 2.
- Creation of a new open source project requires choosing a license that matches your business goals, and preferably use an existing and OSI approved open source license.
- If you want all future derivative work of your contribution to be distributed in source code format, then the GPL is a logical choice.
- If you care contributing to a library, then you might consider the GNU Library or "Lesser" General Public License (LGPL).
- If you want your contribution to be usable within both open source and proprietary (commercial) products, then you might consider the BSD license or the Apache 2.0 license.

4. Train Your Employees

Companies that use and participate in open source projects are highly recommended to provide open source training to their employees. There are specialized companies that offer such educational services or can help your company create and tailor specific open source courses based on your needs. Most common topics covered in basic open source training include:

- General open source concepts
- Open source licenses
- Risks associated with open source software
- Your company's open source policies and compliance rules
- Open source development model
- Integrating open source software within your software development model
- Working with the open source community

Working with the open source community is very different from the traditional corporate development environment and has a different process and set of values from traditional proprietary development model. Training will help bridge the gap and will educate your employees on the working methods of the open source community.

5. Build the Open Source Project Infrastructure

As part of the open sourcing process, you need to build the infrastructure for your project which allows the project to be visible to the outside world and offer communication and software development tools to the project team and to the open source community. A typical project infrastructure includes a web site, a code repository system, one or more project mailing lists, a bug tracking system, a release or patch tracking system and a feature request tracking system.

- *Web site*: The web site of any code contribution must specifically:
  - describe the purpose of the contribution
  - explain the problem the contribution solves
  - explain how the contribution works
  - describe the benefits of the contribution to the common user
  - provide test cases and test scripts so that open source developers can experiment with the contribution and see its benefits
  - advertise news related to the project
- *Code repository system*: Two popular code repository systems are Concurrent Versions System (CVS) and Subversion (SVN). Select one and populate it with your source code. It is also useful to define a coding standard and enforce it.
- *Mailing lists*: You need to host one or more project mailing lists. For instance, your open source project might require a mailing list for developers involved in the project (whether they are your company's developers or external developers from the open source community), another mailing list for the user community, and possibly a third one for quality assurance or software testing. The goal of having more than one mailing list is to keep discussions focused. Typically, the core team of the project would be subscribed to all mailing lists, while other contributors would be subscribed to the list that is of most interest to them.
- *Bug tracking system*: A bug tracking system allows individuals or groups to report bugs against your project; it allows you and others to keep track of outstanding defects.
- *Release or patch tracking system*: A release tracking system allows individuals or groups to keep track of your project software releases or patches.
- *Feature request tracking system*: You should define a process for users to submit feature or enhancement requests and select an appropriate request tracking system.

SorceForge.net offers this entire infrastructure for open source projects, for free. Thousands of open source projects are currently hosted on that site. As an example, you can visit http://sourceforge.net/projects/ppacc/, the web site for a Motorola contribution to the Linux kernel called Precise Process Accounting, and examine all the features provided by the web site.

6. Announce the Project

Once you have created your project infrastructure, you are now ready to announce the project to the world and invite people to provide feedback and contribute to the project. The primary method used to announce your project is to send an email to the relevant mailing lists. Projects can also be announced at conferences, through press releases, or through articles published in the Linux Journal, the Enterprise Open Source Magazine, or any other Linux focused publication.

When making the announcement via mailing list there are some general guidelines to respect:

- Use subject line "[ANNOUNCE] X" to announce the contribution where X is the name of the contribution
- Give some background and introductory text
- Include motivations for your contribution
- Explain how your contributions is different from existing or similar code
- Explain how people will benefit from it
- Don't attach any documents in the email; instead point to a web site.

7. Follow Open Source Development Model and Community Practices

Now that you have announced your project to the world and hopefully have attracted the attention of open source developers, it is your team's responsibility to respect and follow the open source development model and open source best practices. Below we outline the most important practices:

- *Listen to the open source community*: The feedback received from open source developers over the project mailing list can sometimes be negative. Don't worry about it. It is important to review the feedback and understand what the developers are trying to say. It is best not to take such negative feedback personally. After all, the intention is to improve code quality through intensive review. Take the good suggestions you receive and incorporate them into your code. If there are solid technical reasons why the suggestions are not valid then explain those reasons over the mailing list.
- *Embrace code reuse*: Open source developers promote and encourage the development of reusable software. In line with this practice, if someone has already implemented the capability or feature you need, you can use it and build on top of it. In the event that you are starting a new project from scratch, keep software reuse in mind and develop code in modules that can be used by others and by you without many modifications.
- *Be open*: It is important to be open in terms of disclosing problems, bugs and challenges. This openness is much appreciated and is also expected by the open source community, who will help you with immediate workarounds and fixes.
- *Release early and release often*: Open source projects tend to make software releases available early to the user community and then issue frequent updates as the software is modified. This practice is called "release early, release often". The open source community believes that this practice leads to higher-quality software because of peer review and testing by a large base of users who will report bugs and contribute fixes. A side benefit of having many people looking at the code is that the code is reviewed for adherence to coding style; fragile or inflexible code can also be improved because of these reviews. Furthermore, this practice allows the release of small incremental changes that are easier to understand and test.
- *Follow community coding style*: The open source community follows a strict coding style to make it easier to understand the code, review it and revise it quickly.

8. Be Visible

It is important to be active and visible not only when you first launch your project but throughout your project's lifecycle, as this will allow you to rally an increasing number of contributors. You can write articles about your project in open source magazines such as Linux Journal, Linux Magazine, Linux Planet and Open Source Enterprise Magazine; you can attend and present at open source community events and conferences such as the Ottawa Linux Symposium and Melbourne's linux.conf.au conference.

"As part of the open sourcing process, you need to build the infrastructure for your project which allows the project to **be visible to the outside world and offer communication and software development tools to the project team and to the open source community.**

9. Be a Good Open Source Citizen

Being a good open source citizen starts by being part of the open source community, contributing to the community, following and respecting its practices and processes, and leading by example, i.e., getting things done by doing them.

Conclusion

This article reviewed some of the best practices to follow when open sourcing a proprietary technology. Many companies have tried to go the "open source way". Some have bailed after failed trials; other companies are not doing so well and others succeeded and are being regarded as raw models for working with the open source community. It is your responsibility to learn, understand and follow the working methods and best practices of the open source community.

In a follow up article we will describe how we went through the described process to open source a contribution from Motorola to the Linux kernel called the "Precise Process Accounting", which is available today from: http://sourceforge.net/projects/ppacc/.

Stay tuned!

Working with the open source community is very different from the traditional corporate development environment **and has a different process and set of values from traditional proprietary development model.**

AD