How to create a successful (in business and development) open source project. 2016 Computing conference

Michael Widenius
MySQL & MariaDB founder
monty@mariadb.com

Erich S. Raymond & Linus Torvalds

"Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix will be obvious to someone”

or how Linus Thorvalds rephrases this

“Given enough eyeballs, all bugs are shallow”

This may be true for projects like Linux with 12,000 committ
but it's not true for most open source projects

Statement from an open source PDF facility

“I've said it before and I'll say it again:
Good engineers build great technology; great engineers also create a sustainable business model. Based on my 15 years of experience in open source,
I know that it's almost impossible to create a sustainable business model based on the Apache Software License [used by OpenSSL].”

Bruno Lowagie's Online Gazette
http://lowagie.com/heartbleed_asl_bumodel

This talk is for

- People wanting to understand how open source works.
- People wanting to create an open source project.
- People wanting to participate, drive or fork an open source project.
- People wanting to create a profitable company developing an open source project.

Many of the slides are designed to be used as a checklist if you ever want to drive an open source project or conv your company to do more open source.

11 reasons open source is better than closed source

- Better security, auditiablity (no trap doors and more eye balls)
- **Better quality**; Developed together with users
- **No vendor lock in**; More than one vendor can give support
- Better customizability; You can also participate in development
- Using **open standards** (no lock in into proprietary standards)
- Resource friendly; OSS software tend to work on old hardware
- **Lower cost**; Usually 1/10 of closed source software
- No cost for testing the full software
- **Better documentation** and more troubleshooting resources
- **Better support**, in many cases directly from the developers
- When using open source, you take charge of your own future

Note that using open source does **not** mean that you hav
become a software producer!

Open source, the good and the bad

- Open source is a better way to develop software
  - More developers
  - More spread
  - Better code (in many cases)
  - Works good for projects that can freely used by a lot of companies in their production or products.

- It's very hard to create a profitable company **developing** an open source project.
  - Not enough money to pay developers.
  - Hard to get money and investors for most projects (except for infrastructure projects like libraries or deamon services).

Top reasons for creating an open source project

- Solving ones own personal problems
  - For the company you are working (MySQL/PHP)
  - As a research project, part of your hobby (LINUX)
- Re-license an existing closed source project to open source to get the benefits of open source (Netscape).
- To get more developers on an internal company tool/project (no money loss if Open Source)
- Wanting to earn money and at the same time do something good.
- Wanting to give back something to the open source community.
- Wanting to create a company and compete with the guys' with “the power of open source”.

Developing Open Source software is in human nature or why Open Source works

- You use open source because it's less expensive (and re-usable)
- You solve your own problems and get free help and development efforts from others while doing it.
- You pariticpate to increase your reputation (to get a new job?)
- You give patches back to not have to maintain them.

There are of course people that participate because they belive in the open source cause or want to help others, but these are a minority (and most still gets paid for this)

Before starting a new open source project

- Check that if there is already an existing actively developed project.

› It's always better to participate than to do new project or a fork!

- If there are only old dead projects, do some research why they failed and learn from their misstakes.

› Sourceforge, Github and Launchpad is filled with dead projects.

- Find a company or a group of users that wants to work with you to define the scope of the project.

› You want to have users ASAP that are using the project in production!

It's not just software

- The most important part with open source is to create an active community!
- You also need (to interact with the community):

› Web pages (someone needs to design these)
› A forum or a knowledgebase, email lists, bug system.
› Documentation & localization
› Packages, build system, mirrors (for downloads)
› Source code repository

In the beginning you can use github or launchpad to host the project and Open Build Service to build your project, but over time you will need more controll and do this yourself.

You need a good team and active community

- A designated active leader (Linux) or active leaders (MariaDB / PostgreSQL) that have the respect from the community.
- A group of good open Open Source Citizens maintaining the community.
- People with good "old time" coding style standards that actively teach others and participate with the community.
- Active and passionate user and developer community around your product.
- Developers that is using the product daily in production.
- Developers that need to extend the product for their own needs.

Generally the quality of the product is defined by the technical leaders

Transparency is critical for long time success

- An open development model (all email list, discussions architecture, decisions should be open).
- Clear guidelines for how things are done and will be done.
- Clear license and business model.
- Extensive documentation.
- Be clear about your roadmap and release schedule.
- Good open review process of patches that enforces quality.
- Be transparent with your plans and let users influence them.
- Be open about your bugs, and when they will be addressed

.

Keep your promises!

Communicate with your community

- Attend conferences and talk about your product.
- Listen to what your users want and either do it or help them do it themselves.
- Make it easy for people to communicate with you and find information about your product!
- Ensure that most questions on your forums and email lists are answered.
- Understanding that people who write the code 'owns' the code
- Don't expect the community to produce any code.

Recognize that the path for turning a user to a custom… can be long, and don’t let short term monetary interes… distract your long term winning strategy

Be good open source citizens

- Be open about your use of external code - Share the fame!
- Provide quick feedback to providers (bug reports and fixes, code enhancements, ideas)
- Help others in forums (around your code, and around related code you know)
- Keep up a high ethical standard on the email lists/forums.
- Don't talk bad about competing projects.

Fact based comparision is ok.

- Being a good Open Source Citizen will build trust in you and your product, and will pay off over time.

Building a supportive Community later in the gam extremely hard - You should start day 1

Release early - release often

- This will allow your users to:
  - Participate in the development early (many wants to!)
  - Give you comments about your architecture. This is especially important when participating with another project.
  - Give you early feedback for the features that the community would like to see in what you plan to do (often things you haven't thought about)
  - Do early testing; Developers are prepared to test alpha code if the project is interesting!
  - Participate in your project instead of doing a compeating patch / project

Don't be afread to release code with bugs.

It's perfectly fine to have some bugs in Alpha code!

Make it easy to get and use your product!

- Release both binaries and source (if it's your project)
- Aim that each release, including alpha, should be bug free enough to be usable in production by those that try it.
  - People will first use your product and only start extending when they belive in it and it's easier to extend it for their own needs then move to something else.
- Work with the distributions and cloud providers to get the product in there.
- Ensure that the product compiles and works on all major platforms
- Use a good open toolset for development (Sourceforge, Launchpad, github or own) to make it easy for others to participate.

The cost of open source products is not 0

If you are depending on an open source products in your business, you should do your best to ensure it surives:

- Take support from the company producing the software
- If there is a missing critical features you need:
  - Create a team to develop them, together with the community and contribute the feature back to the project
  - Contract the original developers to create the feature
- Do not wait for someone else to get it developed, as it's likely to never happen or happen too late for you!

If you are not contributiong in some form, the open sour product may never evolve in the direction you want!

Success/State of Open Source

- Apple wouldn't exist without open source
- Most servers are running an open source operating system.
- Much of the new technology, like solutions for big data, are open source.

In 2007, before Iphone and Android, we said “**Open Source is winning**, it's only the applications left”

Desktop is getting better, but most new applications for mobile are **closed source**:
1.6M apps in Google play, 1.5M apps for iPhon
F-droid has only **1,700** open source apps

Doing business with Open Source

Open Source is a philosophy and a development model. For some people even close to a religion.

Open Source does **not gurantee** that you will get enough money to develop and/or support your product.

The only open source companies that's done an IPO are RedHat, SourceFire and Hortonworks.

Different kind of Open Source products

- Products developed by the community.
  - Typically larger projects that are developed by many companies to solve their business critical needs or to embedd in their products.
    - Linux, PHP, Apache, MariaDB
- Products that are a tools for the company and released as open source to get more community development.
- A small team or company that is developing and driving an Open Source product as their main offering.
  - These have often a hard time to compete with closed source software (not enough money for full time developement).
    - MySQL, JBoss, Wordpress, most open source software

Questions to consider when creating a company

- Do you plan for a virtual company (“no offices”)?
- Are you creating a company of equals? Should the company be owned by employees (Hacking business model)?
- Do you want to concentrate on services or development ?
  - If development, what license to choose?
- Do you plan to have a big community or work with a few big companies ?
- Do you plan to take in investors ?
  - If yes, then you need an 'exit' plan.

What business model to choose?

- Service company
  - Man powered services (support, training, consulting)
  - Valuation 2 x revenue
- Software company
  - Licensing, Software as a service (SAAS), Subscription
  - Valuation 10 x revenue + X * number of users

To succeed “big time” you need a way to force some of your users to pay!

What is the final goal with the company

- Sold on the market
  - Unpredictable future for product/employees
  - Quick, often high, profit for owners
- Go public on stock market
  - Original owners can still be part of steering the company
  - Somewhat unsure profit (as owners can typically only sell after 6 months)
- Owned by founders, employees (and investors)
  - Stable predictable future
  - Owners gets dividends, employees gets bonus
- Create an open source foundation

Why go Open Source?

- Spread the product more quickly (more users)
- Get some part of the development done elsewhere (lower cost)
- Get things more tested and more bug reports (higher quality)
- Possible to get development done in 'not business critical' directions (more useful product)
- Easier to find good developers, partners and customers

- The above means more market recognition, feedback, leads, business, partners and sell opportunities and a strong trademark.
- In general open source projects gets more feedback and better bug reports than closed source project.

Reasons for users to trust a open source vendor

- Open source vendors are more trustworthy as they depend on trust to survive
- No vendor lock in. Your investment in using the product is safe even if:
  - Vendor goes out of business
  - Vendor would surprisingly change business terms radically
  - Vendor would stop supporting the version of the product you are using
- If this would happen to a popular product, someone would go away with the code and start maintain it themselves
- Little risk for hidden trap doors as one can examine the product code

Benefits for developers using open source vendor

- Easy to get access, look at and use the code
- Freedom to examine and change any part of the code to satisfy your business reasons, fix bugs or port to other systems.
- Freedom to find anyone to do the above
- Freedom to use (read, build and change) the code and redistribute the code in an open source environment.

Benefits for big business / countries using open source

- You can develop your own infrastructure to tune the software for your own means (language, unique requirements)
  - Facebook, Google, Booking.com
  - Brazil, Iceland
- You get internal knowhow you can utilize for business
- Not depending on an outside vendor (for minor things).
- No license costs; Low to very low cost of ownership!

- It's in your interest to collaborate with the original community for long term sustainability
  - A full fork is expensive in the long run!
  - You want the original project to survive!

When go Open Source?

IF you can create a sustainable business model around Open Source, proprietary vendors will have hard to compete with you

It's nowadays increasingly harder to sell closed source solutions in a lot of business sectors.

Nor surprisingly, it's hard to choose the right license for your software as it significantly impacts your strategy, i.e.

- How you can co-operate with your user community
- How you can build your business

How to choose an OS license?

Key Questions:

- What is your business idea around the open source product?
  - Services, subscriptions and/or licensing ?
- What rights do you keep to your code when used, modified and potentially redistributed?
- What kind of community do you want around the product?

The main open source licenses (Very simple view)

- **Public domain**
  - Gives user freedom to do anything, including changing your copyright and claim they wrote it.
- **BSD/Apache**
  - Gives user freedom to full use, but needs to keep copyright in the source code.
- **LGPL**
  - Gives user freedom to use freely, but if they distribute it changed, they need to publish the changes under LGP
- **GPL**
  - Gives user freedom to use it freely, but if they distribute it, they need to publish changes and their code under GPL
- **AGPL** (Optional addition to GPL V3)
  - Free usage, but users needs to publish the code and the cod connected to it even if not distributed (like web server).

Business models to use with open source

- **Open-Core Model** - have an open-source core and sell closed-source features on top of it (e.g. SugarCRM)
- **Dual Licensing Model**- one product/project that gets licensed with a viral, GPL-style license and a commercial closed-source license (e.g. MySQL)
  - Another option is "**Business source**" or **Time Delayed Open Source**
- **Services Models** - where you get to download a productized version of an open-source project and pay a fee for the support you get on it for new features.You can normally also pay for training, features etc
- **Subscriptions** (Usually a combination of support, extended product lifetime and guaranteed updates)
- Having a **non profit foundation** fund the developmen

Open Core

- Probably most popular way nowadays for business trying to do open source first time.
- What Oracle is doing with MySQL and EnterpriseDB with PostgreSQL
- Is **not an open source business model**, because it uses closed components and most of the benefits open source developers expect from the product is gone:
  - You can't change, modify, port or redistribute the code
  - You are locked to one vendor
- You may be able to create a small developer community around the product but mainly by people that doesn't need the closed source extensions.
- For community developers, the “worst” possible offer is open core or closed source that used under subscription and stops workin when subscription runs out

Donations and crowd funding

- Crowd funding can be a good way to start a project, but not as good for continues developing of a product. (Hard to get money later for new features)
- Donations can work for big projects, like Wikipedia, but is very hard for smaller projects
  - MariaDB has got less than 5000 euro during the last 5 years in public donations (paypal etc)
  - Most support has come to the MariaDB foundation trough membership fees.
- Advertisment only works for VERY popular web sites and can't sustain a large development organisation

Dual licensing

- Used first by Ghostscript. MySQL was the second product to use it.
- Can only be used when you have full rights to all the code.
- Give out the same code under two licenses, for example GPL or AGPL and normal commercial closed source.
- Companies that can't use the GPL (because they don't want to give out their code) can buy the closed source version.
- GPL only works well for infrastructure, easily embeddable products, like libraries or databases.
- AGPL can work with projects that is used in SAS environment, but only if the company does closed sour changes to the product (Example: iText, mongodb)

Business Source License (or Delayed Open Source)

- Not an open source license, but gives the users similar advantages.
- Source code is available from the start. Anyone has the right to copy, modify & distribute but can't use it commercially under some conditions you define.
- After X years the code automaticly converts to some open source/free license. The date has to be explicitly set in all source code files to avoid misunderstandings
- **Better than Open Core** as this removes the “one vendor” problem and the code will eventually be free.
- **Investor friendly** (as there is a way to force some users to pay).

Wikipedia has a totally different definition of delayed open source

Reasons for users to trust a open source vendor (again)

Benefits the Business source license gives it's users are almost same as for open source:

- No vendor lock in
  - Anyone can develop or support the product.
- All source code is available
  - Anyone can fix bug or add new features.

The “only” caveta is that if you are using the BSL version in **serious** production, you **have** to pay for a limited time

Business Source license
Forces good behaviour

- You have to do frequent releases
- You have to innovate and fix bug, to ensure that users want to have the latest release.
- Ensures that the company is not bought by a 'bad entity' that just wants to close down the project.

This is ensured by that if the project is not developed future, anyone can continue with the original code and, after a short delay, take over the project.

Recommendations for Business Source license

- 3 or 5 year before it becomes open source.
  - Depends on how long you have between releases and how good your investors are.
- Target that 1/100 or 1/1000 should have to pay.
  - The bigger your 'free community' is, the more benefit you get from it.
- Free version and commersal version should have identical source code and tools.
  - Do **NOT** try to combine this with open core or closed source. Instead tune the Business Source license so that it can generate the income you need to succeed.

Business Source license
Resources and background

- Created around 2004 by David Axmark and Monty
- First news article about it (with lot's of comments):
  - http://www.zdnet.com/open-source-its-true-cost-and-where-its-going-awry-by-monty-widenius-7000016024/
- Published in http://timreview.ca/article/691
- Overview at http://monty-says.blogspot.com
- Fred Trotter's “Open Source Eventually License”
  - http://projects.opensource.org/pipermail/license-discuss/2013-August/001072.html
  - https://github.com/ftrotter/OSE
- Fair Source license is a bit similar: https://fair.io

Business Source license
Lots of confusion

- Discussed on Slashdot:
  - http://developers.slashdot.org/story/13/06/26/1552215/monty-suggests-a-business-friendly-license-that-trends-open
- Lots of confused people about the intention:
  - http://readwrite.com/2013/05/31/mysql-co-founder-wants-you-to-pay-up-for-open-source#awesm=~ofq8ilsT7Ja6kG
  - http://channel9.msdn.com/Forums/Coffeehouse/Business-source-Is-it-really-open-source-at-this-point-Monty-Widenius

Business Source
Projects using it

- MaxScale proxy from MariaDB Corporation
  - **https://mariadb.com/bsl**
  - **https://mariadb.com/bsl-faq-adopting**

Some early adoptors, who tried BSL
(BSL is still new and need time to get adopted)

- Baasbox used it early, but switched to a SAS
- RapidMiner used it early, but switched to AGPL
- RhodeCode used it early but switched to community and enterprise (closed source) editions.

Dual licensing and the Business source license
What to do about companies that cheats?

- The base of dual licensing is that a few % pays, rest is using it for free.
- There will always be companies that tries to missuse any license. This doesen't matter!
- What matters is that **no serious enterprise company** will cheat on the license as the implications if they are caught are too big.

The importance of selling licensing

- MySQL would never have been possible without licenses.
- Very hard to get companies to pay for support, sponsorships or development
  - Companies expect to use open source for free
  - Exception is companies you work with to develop the open source project
- Subscriptions are good, but also hard to sell.
  - Investors are not excited by “only subscription business”
- Licensing is “free money” for the project
  - You need one entity that holds the copyright to the whole project or use SAAS (Software as a service)
  - You need to be able to dual license your project
  - Your project is an infrastructure project that is usually embedded into others.

What made MySQL successful?

- We where using it (for data warehousing and web)
- Internet was new and everyone needed a web-optimized DB
- “Virtual company” made it easy to find good people
- New “free” license scheme (this was before Open Source)
  - Free for most, a few have to pay
  - Second program (ghostscript was first) to use dual licensing, MySQL first to do it with GPL.
- Very easy to install and use (15 minute rule)
  - Released source and tested binaries for most platforms
- Friendly and helpful towards community
  - I personally wrote 30,000+ emails during the first 5 years to help people with using MySQL
- Waited with investments until product was “good enough”

Needed, stable and easy to use product with right pric

The end