EOSA MASTERCLASS #1

European
Open Source
Academy

Building and Sustaining Open Source Impact

How open source business owners and corporate users can drive the change.

Amandine Le Pape

Funded by
the European Union

European Open Source Academy

Lesson #2: Foundational Elements for the Success of an Open Source Business

Masterclass Overview

Lesson #0 – The Story of Matrix and Element

Lesson #1 – Recognising the Open Source Opportunity

Lesson #2 – Foundational Elements for the Success of an Open Source Business

Lesson #3 – Leveraging the Right Enablers

Lesson #4 – Finding Business Models That Work

Lesson #5 – Navigating the Corporate Buyer's Dilemma

Lesson #6 – Driving Open Source Impact through Procurement and Policy

Foundational Elements for Success

- Open source is powerful, but there are **many ways of implementing it which may or may not lead to success**.
- Let’s go for a **quick overview of some key decisions to be made to ensure the project starts on the right foot**: from voicing the clear objectives, to avoiding the trap of open washing, via an analysis of the actual costs of building open source.

But what is open source?

Defining Open Source

- **The definition of open source may be taken at face value as:**

*Software (or hardware) for which the original source code (resp. design) is made freely available and may be redistributed and modified.**

- However, if one looks at the OSI's definition of it, one will find that **open source is not just about being source available, but also about defining user freedoms**.

* Source: Oxford Languages dictionary

OSI's Open Source Definition (Summarised)

- **Free Redistribution** - Licenses must allow software to be given away or sold without requiring a royalty or fee.
- **Source Code Availability** - The program must include source code, allow distribution in both source and compiled forms, and not permit obfuscated source as a disguise.
- **Allow Derived Works** - Modifications and derivative works must be allowed, and they must be distributable under the same license as the original.
- **No Discrimination & Licensing Uniformity** - Licenses must not discriminate against persons, groups, or fields of endeavor, and the rights must apply uniformly to all recipients without extra licensing steps.
- **Neutral & Non-restrictive Terms** - Licenses must not be specific to a product, technology, or restrict other software distributed alongside, and must be technology-neutral.

* *Source*: *https://opensource.org/osd*

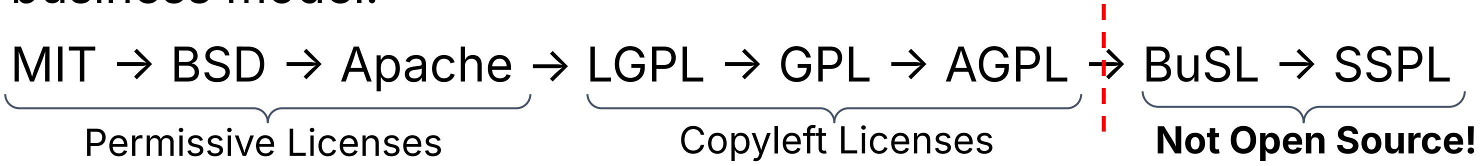
One definition but a range of permissions

- The world of open source licenses is a busy one, but whilst there are hundreds of them they are not all equal.
- Even within the subset officially recognised by the Open Source Initiative, which includes ~80-90 licences
- They can be split into two main groups: **copyleft and permissive**
- Copyleft licence: anyone has the right to use, modify, and share the work, as long as it is open source. The original author can claim copyright on it otherwise. Whether this applies to all of their code or just the modifications they've made to the licensed code depends on the license.

* Source: https://opensource.org/osd

Open Source v. Source Available

- A common temptation is to declare your software “open source” just because you make its source code available somehow.
- The user rights & freedoms are a critical aspect of open source however: the right to extend and redistribute without fees - this is what provides actual sovereignty.
- You can think of this as a “rights ratchet” ranging from most permissive to most restrictive - where the Open Source Definition draws the line in the sand.
- Open source businesses need to find the right equilibrium point for their business model.



MIT → BSD → Apache → LGPL → GPL → AGPL → BuSL → SSPL

Permissive Licenses

Copyleft Licenses

Not Open Source!

Open Source vs. Faux-pen Source

- As we have seen in the first section, **there is a real opportunity** for building an open source business today.
- Which also means that **some may use it as a marketing tool** and try to check the open source box to make them look better.
- Whether you call it **“open washing” or “faux-pen source”**, it presents a few key traits.
    - Source code will be open and available (**source-available**), but:
    - No community
    - No documentation
    - No comments on the code, which will be hardly understandable
    - No ecosystem of vendors, hosters, consultants
    - A very restrictive licence
    - Minimal features in the open source version, making it barely usable, most others being locked in a proprietary fork

The Players of Open Source

There are three fundamental archetypes in open source activities:

- **Non-commercial hobby** projects and volunteering (e.g. Home Assistant)
- **Non-commercial professional** projects outside the critical business path of the driving entities (e.g. React, Kubernetes)
- **Commercial professional** projects on the critical business path of the driving entity (e.g. Element, Nextcloud)

As the focus of this lesson is open source businesses, we will focus on the latter for now.

Foundational Elements for Success

The Challenge of Building an Open Source Business

- Open source businesses can be **built from different positions**
  - As the founding team of an open source project
  - As a contributor to an existing open source project
- But overall **they face the same challenges, albeit with different emphasis**
- And to be clear: these challenges make **building an open source business is even harder than one built purely on proprietary software**, as there face the same moving parts, plus new ones on top:
  - Building on a moving baseline with contribution from multiple origins that you don’t control
  - Dependent from the project’s core team
  - Harder to differentiate against other companies building on top of the same open source baseline
  - Need to take the community into account
  - Need to decide what software is contributed vs kept as a closed source differentiator (if any)
- So, you need to make sure you **get the foundations right**, to be able to focus on the next level up.

Understanding the Real Costs

- The fact you are **building on top of an open layer does not mean you can shave off 100% of the cost of the effort** of developing and maintaining it, on the contrary.
- However, the **cost profile can be different** depending on whether you are building on a project you control, or a project you contribute to.

⇒Let’s look at both situations.

Understanding the Real Costs

When Commercialising a Project You Control

All in all, **it will probably cost you more to build a (real) open source business than a proprietary one**. In particular if it is one that you control because you hire the core team.

Here is why:

1. **You can't cut corners**
   - Developing a product in the open means you are under scrutiny and need to build it in a way others can understand and contribute to.
2. **You need to review & maintain external contributions**
   - The community will contribute to the project, which means your team will have to review their contributions, support the developers and eventually maintain the contributions.
3. **You need to support the external community as well as users**
   - Even if you don't pursue a business model where you need as many people to build on your project (e.g. with SDKs being your core product), you will need to support and interact with the community at some level.
4. **You should to provide documentation**
   - Whilst maybe not a priority, if you aim for the community to use your project, so you need to provide clear documentation, and it takes time

Foundational Elements for Success

European Open Source Academy

Understanding the Real Costs

When Commercialising a Project You Contribute To

The impact is lessened if your **business stands on a project controlled by another outfit or the community**. There are nonetheless additional costs compared to a proprietary business:

1. **You need to keep up with the changes made to the project**
   - That may include software changes impacting APIs you integrate with, new features, technology shifts meaning your team needs to train on something new...
2. **Your contributions to the project take time to land**
   - The team controlling the commits to the project becomes a bottleneck
3. **You should want to financially support the project on top of which you are build:**
   - It is good practice to not freeride and actually support the project you are building upon.
   - This can be done in different ways depending on how the project is set-up: it can be a membership to the Foundation running the community and the core repositories, or using the paid products / buying services from the commercial vehicle of the core team, or code contributions but they would need to come with maintenance.
4. **It is expensive to build up code expertise to provide deep level of support**
   - You might struggle to provide a deep level of support as you don't have code ownership, so you would need to ramp up if that is relevant to your business model.

Strategic Control Points

In short, before you start, make sure you are clear on:

1. What benefit you are trying to get from building an open source business: is it marketing or the real deal?
2. What are the new constraints this brings, in particular for a “real” open source business?
3. What additional costs are brought by these constraints?

Summary

In this lesson, we have:

- Agreed on the definition of open source when used in this discussion, and introduced warnings about open washing.
- Looked at the challenges of building an open source business
- Looked at the costs that one might ignore
- Provided a list of key questions to ask oneself before starting.

EOSA MASTERCLASS #1

European Open Source Academy

Building and Sustaining Open Source Impact

How open source business owners and corporate users can drive the change.

Amandine Le Pape

Funded by
the European Union