Open Source Software Paradigms

Lecture - 02

Economics and Business Models of Open Source

Lecturer: Biniam Behailu
Addis Ababa Science and Technology University

Learning Objectives

By the end of this session, you will not just understand that open source is commercial, but how and why.

- ☞ You will be able to articulate the fundamental economic paradox of OSS and its resolution through the lens of value creation versus value capture.
- ☞ You will be able to categorize, compare, and critique the dominant modern revenue generation models.
- ☞ You will analyze the diverse funding mechanisms that sustain projects, from venture capital to foundations and crowdfunding.
- ☞ You will deconstruct the strategic motivations for corporations to invest billions in open source, beyond direct revenue.

Contents

- ☞ Closed Source / Proprietary / Free / Open Source Software
- ☞ The Economic Paradox in Open source
- ☞ Economics of Open Source Software
- ☞ Open Source Business Models

Closed Source / Proprietary / Free / Open Source Software

***Image source**: Timrawi, T. (2023, October 3). What Is Open Source & Closed Source? Sharktech. [Accessed September 15, 2025], from https://sharktech.net/what-is-open-source-closed-source/*

Closed Source vs Proprietary Software

- Closed-Source and proprietary software are often used interchangeably, but they do have slight differences.

Closed Source Software

- ☞ Software whose source code is **not** made **available** to the public.
- ☞ The development **process**, **codebase**, and **inner workings** of the software are kept **private** and typically owned by a single entity or organization.
- ☞ Users of closed source software can only access the **compiled version** of the software, usually in the form of **executable binaries**.
- ☞ They do not have the ability to **view** or **modify** the underlying source code.

Closed Source vs Proprietary Software

Closed Source Software

- **Licensing**: Closed source software may be free to use but still does not provide access to the source code.
- **Free to use Software**: Some software may be closed source but free to use, such as certain versions of Adobe Reader (which is free but not open source).
- **In-house Software**: A company may develop software for internal use that is closed source but not sold or licensed to others.

Closed Source vs **Proprietary Software**

Proprietary Software

- Owned by an individual or company, and its usage is restricted through licenses.
- The term proprietary signifies that the **rights** to the software, including its **distribution**, **modification**, and **use**, are restricted and protected by copyright or other legal mechanisms.
- **Licensing**: Proprietary software typically requires a paid license and comes with restrictions on how it can be used, shared, or modified.
- Proprietary software often involves licensing agreements that dictate how the software can be used, distributed, and modified.
- It also prevents the user from copying the software, modifying it, or sharing it with associates and friends.

Closed Source vs **Proprietary Software**

Proprietary Software

- Users must typically agree to these terms before they can install or use the software.
- **Commercial Software**: Microsoft Office and Adobe Creative Cloud are proprietary, requiring users to purchase licenses.
- **Subscription Services**: Software as a Service (SaaS) products, like Salesforce, are proprietary and often charged on a subscription basis.

Closed Source vs **Proprietary Software**

Proprietary Software

- **Access to Source Code**: Most proprietary software is closed source, and not all closed source software is proprietary.
- For example, if a company develops an internal tool that isn't sold or licensed, it is closed source but not proprietary.
- **Usage Rights**: Proprietary software imposes more restrictions on how the software can be used and distributed compared to some closed source software that may be freely available for personal or educational use.
- Software can be distributed with various options like Freemium, Dual licensing, Paid software, Freeware, Trialware etc…

Closed Source vs Proprietary Software

Note

- While closed source software is often proprietary, there are instances where proprietary software can be open source.
- **Dual Licensing**: Some companies release their software under both a closed source proprietary license and an open source license, allowing users to choose which license they prefer.
- This model enables companies to monetize their software while also providing an open source option for those who prefer it.
- **Example**: MySQL, MongoDB, Qt Framework, Redis, ElasticSearch

Closed Source vs Proprietary Software

Note

- **Source-Available Software**: Some proprietary software is released with access to the source code, but usage rights are still restricted by a license agreement.
- This model allows users to view and modify the source code for personal use or evaluation purposes, but they may not be allowed to redistribute or modify the software for commercial purposes.
- **Examples**: Oracle JDK (Java Development Kit), GitLab Community Edition, Xamarin

Closed Source vs Proprietary Software

Note

- **Shared Source Initiative**: is a licensing model introduced by Microsoft that allows users to access the source code of certain Microsoft products under specific conditions.
- While this allows for greater transparency and collaboration, it still retains proprietary control over the software.
- **Examples**: .NET Framework, Windows Internals

Free vs Open Source Software

- **Open source** software and **free** software are closely related concepts, but with minor differences in **philosophy** and **focus**.
- As described in chapter one, **free** Software emphasizes **user freedom** and **control** over the software they use, **not price** [1] .
- Supported by the **Free Software Foundation** (FSF) and **Richard Stallman**.
- It's a moral and ethical imperative. The four essential freedoms are: use, study, modify, and share.
- The license (**GPL**) is designed to protect these freedoms forever ("**copyleft**").

Free vs Open Source Software

- Focuses on four essential freedoms users should have with software:
  1. The freedom to **run** the program for any purpose.
  2. The freedom to **study** how the program works and modify it.
  3. The freedom to **redistribute** copies so you can help your neighbor.
  4. The freedom to **improve** the program, and **release** your improvements to the public so that the whole community benefits.

Free vs **Open Source Software**

- Open Source Software emphasizes practical **collaboration** and **development** to create better software
- Focuses on the benefits of open source code for building quality software.
- Open source software must have its source code publicly available under a license that respects:
  - ☞ Free **distribution**
  - ☞ Ability to **modify** the source code
  - ☞ Ability to **distribute** modified versions

Free vs Open Source Software

- The difference between free and open source in

Philosophy

- ☞ Free software advocates for **user rights**
- ☞ Open source focuses on **development** advantages.

Source code

- ☞ Free software doesn't necessarily require open source code, but most open source software is also free software.

Free vs Open Source Software

Criteria	Free Software	Open Source Software
Philosophy	User empowerment and control	Collaborative development for better software
Focus	User freedoms	Practical benefits of open source code
Source Code Availability	May or may not be publicly available	Must be publicly available under a specific license
Key Advantage for Users	Freedom to use, modify, and distribute	Access to source code for better security, customization, and potential contribution

Free and Open Source Software (**FOSS**)

- FOSS stands for Free and Open Source Software, encompassing both free and open source software.
- It refers to software that respects **user freedom** and is **open source**, meaning that the source code is made available to the public [2].
- Represents the majority of software that is both free to use and has open source code.
- OSS promotes sharing, collaboration, community engagement, transparency, and the ability to customize software for individual or organizational needs.
- Some well-known examples of FOSS include the **Linux OS**, the **Apache HTTP Server**, and the **Mozilla Firefox** browser.

Free vs Open Source vs FOSS

Term	Focus	Key Characteristics	Example
Free Software	User Freedom	Four essential freedoms: run, study, modify, and distribute. Source code availability not mandatory.	• GNU Image Manipulation Program (GIMP) • LibreOffice (with some versions not fully open source) • VeraCrypt (disk encryption)
Open Source Software	Collaborative Development	Publicly available source code under a permissive license for modification and distribution.	• Linux Operating System • Apache web server Firefox web browser
FOSS (Free and Open Source Software)	User Freedom & Collaborative Development	Combines philosophies of both. Free to use and modify, with publicly available source code.	• Most Linux distributions (e.g., Ubuntu, Mint)

Open-source vs Source-available

- Both open-source and source-available software allow you to see the code behind the program, with little differences.
- **Open-source grants broad permissions.**
- You can typically use, modify, and distribute the code for any purpose, including commercial use.
- Open-source projects thrive on this collaboration and openness, which can lead to faster innovation and bug fixes.
- There are well-defined licenses, like **MIT** and **Apache 2.0**, that guarantee these freedoms.

Open-source vs Source-available

- **Source-available is more restrictive.**
- While you can see the code, there may be limitations on how you can use or modify it.
- These limitations could restrict things like:
  - ☞ **Field of use**: You might only be allowed to use the code for specific purposes, like personal projects and not commercial ones.
  - ☞ **Modification and distribution**: You might not be allowed to modify the code or redistribute it to others.

This can make source-available software less appealing for developers who want to tinker and improve the code.

Open-source vs Source-available (Key Differences)

Freedoms

Open source grants users the freedom to use, modify, and distribute the software, while source-available doesn't necessarily guarantee these rights.

Licenses

Open source uses specific licenses that uphold these freedoms, while source-available can use various licenses, some with restrictions.

Development

Open source encourages collaboration and community development, while source-available might not.

Open source software is like a public park

Everyone can come in, play, and even help improve the facilities.

Source-available software is like a museum with a glass window

You can see what's inside, but you can't touch or modify it unless given permission.

Economic Paradox in Open source

The Fundamental Economic Paradox

The $0 Price Tag

- In traditional microeconomics, software is a non-rivalrous good (my use doesn't prevent your use), but it is excludable (you can be prevented from using it via licensing).
- This excludability is the basis for traditional software monetization.

***Image source**: Easy-Peasy.AI, "Free Price Tag: $0," 2024. [Online]. Available: https://easy-peasy.ai/ai-image-generator/images/price-tag-0-dollars. [Accessed: Sep. 15, 2025].*

The Fundamental Economic Paradox

The $0 Price Tag

The Paradox

- ☞ Open source deliberately removes **excludability**.
- ☞ It gives away the product of immense intellectual effort, a valuable economic good, for a price of **$0**.
- ☞ This appears to violate basic economic principles of investment and return.

***Image source**: Easy-Peasy.AI, "Free Price Tag: $0," 2024. [Online]. Available: https://easy-peasy.ai/ai-image-generator/images/price-tag-0-dollars. [Accessed: Sep. 15, 2025].*

The Fundamental Economic Paradox

The $0 Price Tag

The Paradox

- ☞ This paradox only exists if we assume the value of software is solely and exclusively contained within the bits of the executable and its direct license fee.

***Image source**: Easy-Peasy.AI, "Free Price Tag: $0," 2024. [Online]. Available: https://easy-peasy.ai/ai-image-generator/images/price-tag-0-dollars. [Accessed: Sep. 15, 2025].*

The Fundamental Economic Paradox

The $0 Price Tag

The Resolution

- ☞ The value is **not just in the code**; it's in the **ecosystem**, the **support**, the **reliability**, the **brand**, and the **market position** it creates.
- ☞ OSS business models are about **creating value** through the **code** and **capturing** value elsewhere [3].

***Image source**: Easy-Peasy.AI, "Free Price Tag: $0," 2024. [Online]. Available: https://easy-peasy.ai/ai-image-generator/images/price-tag-0-dollars. [Accessed: Sep. 15, 2025].*

Value Creation vs. Value Capture of OSS

- Value Creation(the commons): This is how the open source project generates value for the world.

Collaborative Development

Thousands of developers improving code, finding bugs, adding features.

Rapid Innovation

Faster iteration than proprietary models.

Standardization

Becoming a de facto standard (e.g., Linux for servers, Kubernetes for orchestration).

Massive Adoption

Low barrier to entry leads to a large user base, network effects, and a rich ecosystem.

Value Creation vs. **Value Capture** of OSS

- Value Capture (Monetization): This is how an entity (company or individual) captures a portion of that created value as revenue.
- The business model is the bridge between creation and capture.

Selling Scarcity
Features (Open Core), Legal Certainty (Dual License).

Selling Convenience
Hosting, Management, Support (SaaS, Support Services).

Selling Influence
Priority access to roadmap, dedicated resources.

Economics of Open Source Software

Economics of Open Source Software

1. Cost Structure

- **Free to Use**: OSS is typically available at no cost, allowing users to download, use, and modify the software without licensing fees.
- This reduces barriers to entry, especially for startups and small businesses.
- **Support Costs**: While the software itself may be free, organizations often incur costs related to support, maintenance, and training.
- Some companies charge for premium support services.

Economics of Open Source Software

2. Business Models

- **Freemium Model**: Many OSS projects adopt a freemium approach, offering basic features for free while charging for advanced features, support, or services (e.g., Red Hat).
- **Dual Licensing**: Some projects are offered under both open source and proprietary licenses, allowing the developers to monetize the software in specific contexts (e.g., MySQL).
- **Consulting and Services**: Companies can generate revenue by providing consulting, implementation, and customization services for open source software.

Economics of Open Source Software

3. Value Creation

- **Community Contributions**: Open source relies on a community of developers who contribute to the software.
- This collaborative model can lead to rapid innovation and improvement without the costs associated with traditional development.
- **Network Effects**: The value of OSS increases with the number of users.
- As more people use and contribute to a project, its robustness and feature set improve, attracting even more users.

Economics of Open Source Software

4. Market Dynamics

- **Competition with Proprietary Software**: OSS often competes directly with proprietary software, leading to lower prices and improved quality in the market.
- This competition can drive innovation, benefiting end-users.
- **Vendor Lock-in**: OSS reduces vendor lock-in, allowing users to switch providers or modify the software to meet their needs.
- This flexibility can be economically advantageous for businesses.

Economics of Open Source Software

5. Investment and Funding

- **Crowdfunding and Donations**: Many open source projects rely on crowdfunding, donations, or sponsorship to support development.
- Platforms like Kickstarter, Patreon or Open Collective facilitate financial support from users and organizations.
- **Corporate Sponsorship**: Large companies often sponsor open source projects or contribute to their development to align with their business interests, ensuring the sustainability of key tools they rely on.

Economics of Open Source Software

6. Economic Impact

- **Job Creation**: The growth of the open source ecosystem has created jobs in development, support, and consultancy.
- Companies often seek professionals with expertise in specific open source technologies.
- **Boosting Innovation**: Open source fosters innovation by allowing developers to build upon existing projects, leading to new applications and technologies that can drive economic growth.

Economics of Open Source Software

7. Challenges and Risks

- **Sustainability**: Many open source projects struggle with funding and long-term sustainability, which can affect their viability and the ecosystem as a whole.
- **Quality Assurance**: The varying levels of quality in open source projects can pose risks for organizations that rely on them, potentially leading to increased costs for support and troubleshooting.

Open Source Business Models

- Freemium Model
- Crowdfunding and Donations
- Dual Licensing
- Sponsorship and Grants
- Subscription Model
- Open Core Model
- Consulting and Services
- Marketplace Model
- Support and Maintenance

Open Source Business Models

1. Freemium Model

- The software is offered for free with basic features, while advanced features, additional services, or premium support are available for a fee.
- **Example**: Companies like Atlassian (with products like Trello) and Slack offer free tiers with optional paid upgrades for enhanced functionality.

2. Dual Licensing

- The software is available under both an open source license and a proprietary license.
- Users can choose between the free version (with certain limitations) and a paid version with added features or support.
- **Example**: MySQL allows users to access the database under the GPL for free or purchase a proprietary license for additional privileges.

Open Source Business Models

3. Subscription Model

- Users pay a recurring subscription fee for access to the software, which may include updates, support, and additional features.
- **Example**: Red Hat offers its enterprise Linux distribution through a subscription model, providing support and services to businesses.

4. Consulting and Services

- Companies provide consulting, implementation, customization, and training services for open source software.
- This model focuses on expertise rather than the software itself.
- **Example**: Canonical, the company behind Ubuntu, provides consulting and support services for organizations adopting its open source operating system.

Open Source Business Models

5. Support and Maintenance

- Offering paid support, maintenance, and training for open source software.
- This model is often used by organizations that require reliable assistance and expertise.
- **Example**: SUSE provides enterprise-level support for its open source Linux distribution, catering to businesses that need assurance and stability.

Open Source Business Models

6. Crowdfunding and Donations

- Developers and projects seek funding through crowdfunding platforms or direct donations from users and organizations that benefit from the software.
- Example: Projects like GIMP and Blender have successfully used crowdfunding campaigns to support ongoing development.

7. Sponsorship and Grants

- Organizations or individuals sponsor open source projects, providing financial support in exchange for influence over project direction or to ensure its sustainability.
- **Example**: Large tech companies, such as Google and Microsoft, often sponsor open source initiatives that align with their business goals.

Open Source Business Models

8. Open Core Model

- The core functionality of the software is open source, while additional features or components are offered as proprietary add-ons.
- Example: GitLab provides a free open source version of its platform, with premium features available in commercial editions.

9. Marketplace Model

- Creating a marketplace where third-party developers can sell their extensions, plugins, or services related to the open source software.
- **Example**: Platforms like WordPress allow developers to create and sell themes and plugins while maintaining an open source core.

Where is Open Source succeed and Fail?

Successful Domains

- Community Collaboration
- Flexibility and Customization
- Cost-Effectiveness
- Security and Transparency
- Adoption in Enterprises
- Innovation in Emerging Technologies

Unsuccessful Domains

- Sustainability of Projects
- Fragmentation
- Quality Control
- Complexity of Contribution
- Commercial Competition
- Security Risks

Summary

Key Takeaways

- **The Paradox:** Open source software with a $0 price tag is commercially viable.
- **The Resolution:** Value is created in the collaborative, free code and captured through scarce, paid services.
- **Business Models:** Companies monetize by selling services, support, and premium features (e.g., Freemium, Open Core, Dual Licensing).
- **Broader Economics:** The open source ecosystem drives innovation, reduces vendor lock-in, and creates jobs.

Brain Teaser

1. What is the key difference in philosophy between "Free Software" and "Open Source Software"?

A. Free Software emphasizes a $0 price, while Open Source focuses on user rights.

B. Free Software is about collaborative development, while Open Source is a moral and ethical movement.

C. Free Software is a moral and ethical movement focused on user freedom, while Open Source emphasizes the practical benefits of collaborative development.

D. Free Software requires publicly available source code, while Open Source does not.

Brain Teaser

2. What is Dual Licensing ?

A. To ensure the software can only be used for non-commercial purposes.

B. To allow a company to sell a proprietary license while also providing an open source option.

C. To prevent the source code from being redistributed.

D. To require all derived works to be released under the same open source license.

References

[1] Free Software Foundation, "What is free software and why is it so important for society?," FSF, 2023. [Online]. Available: https://www.fsf.org/about/what-is-free-software. [Accessed: Sep. 15, 2025].

[2] FOSS United, "What is FOSS?," FOSS United Wiki. [Online]. Available: https://fossunited.org/wiki/what-is-foss. [Accessed: Sep. 15, 2025].

[3] J. Lerner and J. Tirole, "The Economics of Technology Sharing: Open Source and Beyond," Journal of Economic Perspectives, vol. 19, no. 2, pp. 99-120, Spring 2005. doi: 10.1257/0895330054048678.

Thank you!

"Open source software is a testament to the power of collaboration; it transforms ideas into innovations, empowering individuals and communities to build a better future together."

Lecturer: Biniam Behailu
Addis Ababa Science and Technology University