Open Source Software Paradigms

Lecture - 14

Course Review, Summary and Assessment

Lecturer: Biniam Behailu
Addis Ababa Science and Technology University

Learning Objectives

By the end of this lecture, you will be able to:

- Remember the core philosophies, freedoms, and economic models of Open Source Software (OSS).
- Differentiate between key OSS licenses and their practical implications for use and distribution.
- Evaluate the components of the modern OSS technology stack and their security lifecycle.
- Synthesize course concepts by analyzing real-world OSS scenarios and proposing strategic solutions.

Contents

What is OSS? The Core Idea

Free Software vs. Open Source

"**Free Software**" (Richard Stallman) is a social movement focused on user freedom. The Four Essential Freedoms:

- ☞ Freedom-0: To run the program for any purpose.
- ☞ Freedom-1: To study how the program works and adapt it.
- ☞ Freedom-2: To redistribute copies to help others.
- ☞ Freedom-3: To improve the program and release improvements.

"Open Source" (**OSI**) is a development methodology focused on the practical benefits of transparency and collaboration.

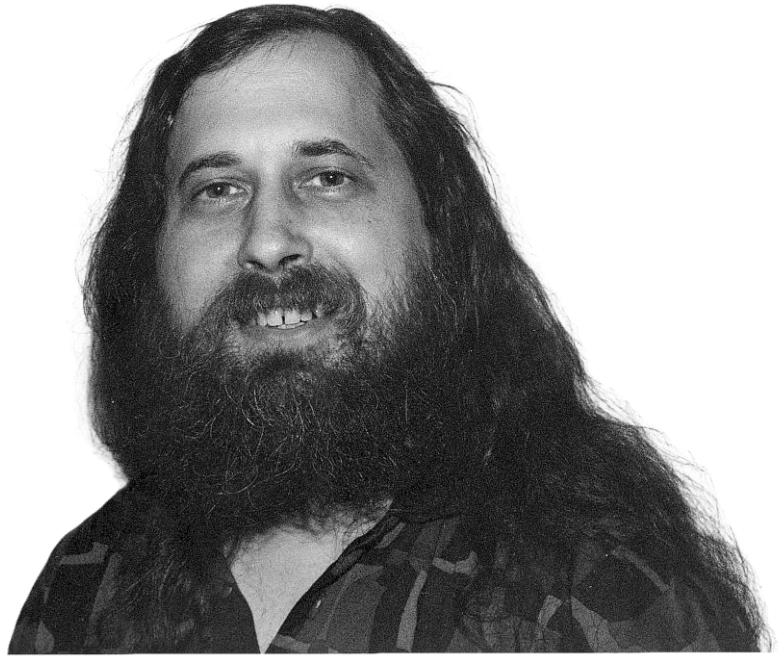

Image source: *"Richard Stallman," unibzmagazine.it, [Online]. Available: https://www.unibzmagazine.it/assets/Events/Images/eng-faculty/Richard-Stallman-800X600__ScaleWidthWzEwNDhd.jpg. [Accessed: 07-Nov-2025].*

History of Open Source

1983: The GNU Project

Richard Stallman launches the GNU Project to create a completely free Unix-like operating system.

1991: Linux is Born

Linus Torvalds releases the first Linux kernel, providing the missing piece for a complete free OS.

1998: The "Open Source" Term

The Open Source Initiative (OSI) is founded, coining the term "open source" to be more business-friendly.

Defining the Paradigms

Aspect	Proprietary Software	Free Software	Open Source Software
Philosophy	Control & Revenue	User Freedom & Ethics	Practical Collaboration & Quality
Source Code	Closed, Secret	Available (Freedom Focus)	Available (Development Focus)
Key Goal	Profitability	Empowerment	Better, More Reliable Software
Example	Microsoft Windows	GNU Emacs	Linux Kernel

The Four Essential Freedoms (Free Software)

- Freedom 0: Run the program for any purpose.
- Freedom 1: Study how the program works and change it.
- Freedom 2: Redistribute copies.
- Freedom 3: Distribute copies of your modified versions.

FOSS (Free & Open Source Software) bridges both philosophies.

The Economic Paradox & Its Resolution

Solving the "$0 Price Tag" Paradox

- **The Paradox**: How can giving away a valuable product (code) for free be sustainable?
- **The Resolution**: Value is created in the open code (commons) and captured elsewhere by selling scarcity.
  - ☞ Value Creation: Collaboration, Innovation, Standardization.
  - ☞ Value Capture: Services, Support, Hosting, Premium Features.

Dominant Open Source Business Models

Model	Mechanism	Example
Open Core	Core is OSS; Enterprise features are proprietary	GitLab, Redis
Dual Licensing	OSS license (GPL) / Commercial license	MySQL, Qt
SaaS/Subscription	Hosted & managed service	Red Hat Enterprise Linux
Support & Services	Paid support, consulting, training	Canonical
Marketplace	Platform for selling extensions/themes	WordPress

The Open Source Definition (OSD)

- The Open Source Initiative defines the Open Source Definition (OSD) as a set of ten criteria that a software license must meet to be considered "open source".

1. Free Redistribution
2. Source Code
3. Derived Works
4. Integrity of the Author's Source Code
5. No Discrimination Against Persons or Groups
6. No Discrimination Against Fields of Endeavor
7. Distribution of License
8. License Must Not Be Specific to a Product
9. License Must Not Restrict Other Software
10. License Must Be Technology-Neutral

- OSD is also called the 10 commandments of OS.

Software Licensing Landscape

- Two Major Open Source License Families
  - ☞ **Permissive Licenses**: Max freedom, minimal restrictions. Easy to use in proprietary software.
  - ☞ **Copyleft Licenses**: Ensure software freedoms are preserved in derivatives.

Permissive Licenses (The "Do Anything" Licenses)

- **Philosophy**: "Do what you want, just give credit."
- Key Licenses: MIT, Apache 2.0, BSD.
- Apache 2.0 is often preferred for its explicit patent grant.
- **Use Case**: Ideal for libraries, tools, and encouraging widespread adoption.

Copyleft Licenses (The "Share-Alike" Licenses)

- **Philosophy**: "You can use and modify, but you must share your changes under the same terms."
- Key Licenses:
  - ☞ **GPL** (Strong Copyleft): Derivatives must be GPL.
  - ☞ **LGPL** (Weak Copyleft): Can be linked with proprietary software.
  - ☞ **AGPL** (Network Copyleft): Triggers for SaaS/network use.
- **Use Case**: Protecting project freedom, preventing proprietary forks.

Intellectual Property (IP) in OSS

- **Copyright**: The legal basis for licensing. Protects the expression of the code.
- **Patents**: Protect inventions and functional processes. Addressed in licenses like Apache 2.0 & GPLv3.
- **Trademarks**: Protect brand names and logos. Controlled separately from the code.

Contributor Agreements

- **CLA** (**Contributor License Agreement**): Formal agreement granting the project necessary rights to use the contribution.
- **DCO** (**Developer Certificate of Origin**): Lighter alternative; a signed-off commit certifying rights.

- **Purpose**: Ensures clean IP chain of title for the project, enabling enforcement and dual licensing.

The Open Source Technology Stack

Open Source Operating Systems

- Linux Kernel: The foundation for most servers, cloud, and Android.
- Distributions (Distros): Ubuntu, Fedora, Debian, CentOS.
- BSD Family: FreeBSD (performance), OpenBSD (security).

The Open Source Technology Stack

Open Source Databases

Type	Use Case	Examples
SQL (Relational)	Structured Data, ACID transactions	**PostgreSQL, MySQL, MariaDB**
NoSQL	Scalability, Flexibility	**MongoDB** (Document), **Redis** (Cache), **Cassandra** (Wide-Column), **Neo4j** (Graph)

The Open Source Technology Stack

Popular Development Frameworks

- **Frontend Web**: React, Angular, Vue.js
- **Backend Web**: Node.js (Express), Django (Python), Spring Boot (Java), Laravel (PHP), Ruby on Rails
- **Mobile**: React Native, Flutter (Dart), Ionic (Web-based)

The Open Source Technology Stack

Big Data Tools Ecosystem

- **Storage**: Hadoop HDFS, Cloud Storage
- **Processing**: Apache Spark (unified analytics), Apache Flink (streaming)
- **Ingestion**: Apache Kafka (real-time event streams)
- **Orchestration**: Apache Airflow (workflow management)
- **Querying**: Presto (distributed SQL engine)

The Open Source Technology Stack

Open Source Office Suites

- **LibreOffice**: The leading, most active community suite (Writer, Calc, Impress).
- **Apache OpenOffice**: A mature, stable alternative.
- **Critical Feature**: Native support for Open Document Format (ODF), an open standard ensuring long-term access and no vendor lock-in.

The Open Source Technology Stack

Content Management Systems (CMS)

- **Traditional** (**Monolithic**): Manages both content and presentation.
  - ☞ WordPress: Powers ~43% of the web. Vast plugin ecosystem.
  - ☞ Drupal, Joomla: Powerful, flexible, for more complex sites.
- **Headless CMS**: Manages only content, delivered via API to any front-end.
  - ☞ Strapi, Sanity. Ideal for omnichannel content.

The Open Source Technology Stack

Enterprise Resource Planning (ERP)

- **Purpose**: Integrated suite to manage core business processes (HR, Sales, Inventory, Finance).
- Open Source Leaders:
  - ☞ **Odoo**: "Freemium" model, modular, user-friendly.
  - ☞ **ERPNext**: Fully open source (GPL), comprehensive features.

The Open Source Security Paradox

- **Strength**: "Given enough eyeballs, all bugs are shallow." Transparency enables community auditing.
- **Weakness**: The same transparency exposes vulnerabilities to attackers.
- **Conclusion**: Proactive security management is essential, not optional.

The OSS Security Lifecycle

A Continuous Process:

1. **DISCOVER**: Create a Software Bill of Materials (SBOM). Know what you use.
2. **DETECT**: Scan dependencies for known vulnerabilities (using SCA tools).
3. **PRIORITIZE**: Use CVSS (severity) and EPSS (exploit likelihood) scores.
4. **REMEDIATE**: Patch, update, or mitigate vulnerabilities.
5. **PREVENT**: Harden CI/CD pipelines with automated security gates.

Common Software Supply Chain Attacks

- **Typosquatting**: requwest instead of request.
- **Dependency Confusion**: Malicious public package overrides internal one.
- **Maintainer Burnout**: Critical project is abandoned, leaving unpatched vulnerabilities.

Development & Contribution

Development Models: Cathedzral vs. Bazaar

Feature	The Cathedral	The Bazaar
Development Style	Centralized, Planned	Decentralized, Collaborative
Releases	Infrequent, Large	Frequent, Small
Governance	Top-Down	Community-Driven
Example	GNU Emacs	Linux Kernel

How to Contribute to OSS

It's More Than Code!

- **Code**: Bug fixes, new features, tests, refactoring.
- **Non-Code**: Documentation, translation, design, community support, triaging issues.
- Start with: **CONTRIBUTING.md** and **"good first issue"** labels.

The Standard Git Workflow

The contributor's journey:

1. Fork the repo on GitHub.
2. Clone your fork locally.
3. Create a Branch for your feature/fix.
4. Commit your changes with a clear message.
5. Push to your fork.
6. Open a Pull Request (PR) for review.
7. Iterate based on feedback.
8. Merge! Celebrate your contribution.

Best Practices for Contributors

- **Communication is Key**: Be respectful, professional, and clear in issues and PRs.
- **Start Small**: Begin with documentation or a simple bug fix.
- **Read the Docs**: README.md, CONTRIBUTING.md, CODE_OF_CONDUCT.md.
- **Embrace Review**: Code review is a learning opportunity, not criticism.

Open Source AI: New Challenges

- Model Licensing: Is an AI model "source code"? What license applies (e.g., Meta's Llama)?
- Training Data: Is using OSS code for training "fair use" or a derivative work?
- Output Compliance: If an AI reproduces GPL-licensed code, who is liable?
- Enforcement: Traditional compliance tools are adapting to scan AI-generated code.

Summary

- **Philosophy & Freedom**: OSS is built on a foundation of user freedom (Free Software) and collaborative pragmatism (Open Source), guided by the Four Essential Freedoms.
- **Economics & Models**: Sustainable OSS separates value creation (the open commons) from value capture (services, support, premium features) through proven business models like Open Core and SaaS.
- **Licensing is Law**: Understanding the distinction between permissive (MIT, Apache) and copyleft (GPL, AGPL) licenses is critical for legal compliance and strategic use.
- **A Stack for Everything**: A robust, enterprise-grade OSS stack exists for every layer of modern technology, from OS and databases to AI and Big Data tools.
- **Security is a Process**: Leveraging the "many eyes" advantage requires a proactive security lifecycle: Discover, Detect, Prioritize, Remediate, and Prevent.
- **Contribution is for Everyone**: You can contribute through code, documentation, design, and community support.

Final Recap Question

Question: A company uses a GPLv3-licensed library in their web application, which is provided as a SaaS. What is their license obligation?

A) They must pay a fee to the original author.

B) They have no obligations because it's a service, not distributed software.

C) They must release the source code of their application under GPLv3.

(Answer: C, because the AGPL-like provisions of GPLv3 can be triggered for network use.)

Course Project

Choose ONE of the following scenarios
and provide a comprehensive analysis.

Option 1: The Startup Dilemma (Beginner - Intermediate)

Scenario:

"TechSolve," a new startup, is building a project management SaaS application. They want to use open-source components to speed up development and reduce costs. They are considering using:

- **PostgreSQL** (Database)
- **React** (Frontend)
- **Django** (Backend)
- **Redis** (Caching)

Your Task:

1. **License Analysis:** For each component, identify its license type (permissive/copyleft) and explain the obligations for TechSolve.
2. **Business Model:** Propose a viable open-source-based business model for their SaaS product.
3. **Risk Assessment:** What is the single biggest risk of building their core product on these OSS components, and how can they mitigate it?

Option 2: The Enterprise Migration (Intermediate - Advanced)

Scenario:

A large university currently uses proprietary software for its entire IT ecosystem (OS, Office Suite, ERP, CMS). Facing budget cuts and vendor lock-in, the CIO wants to evaluate a full migration to open-source alternatives.

Your Task:

Create a High-Level Migration Proposal that includes:

1. Technology Stack: Recommend specific OSS replacements for the OS, Office Suite, and CMS. Justify each choice based on the university's needs (cost, ease of use, community support).

2. Total Cost of Ownership (TCO) Analysis: List the top 3 categories of hidden costs the university must budget for, beyond software licensing.

3. Adoption Strategy: Outline a 3-phase plan to ensure faculty and student adoption, addressing the major challenge of change management.

Option 3: The Fork in the Road (Advanced)

Scenario:

A large university currently uses proprietary software for its entire IT ecosystem (OS, Office "OpenAIHelper" is a critical open-source library under the GPL v3 license, widely used in the AI community. The lead maintainer has abandoned the project. A large cloud provider (CloudCorp) has started offering a proprietary, hosted version of OpenAIHelper without contributing back.

Your Task:

You are a community leader. Draft a Community Action Plan that addresses:

1.The Fork: Should the community fork the project? If yes, what must they consider regarding the project's name (trademark) and governance model (Cathedral vs. Bazaar)?

2.The License: Should the forked project keep the GPL v3 license, or switch to a more permissive license like MIT or a more restrictive one like AGPL? Analyze the pros and cons for the community.

3.Sustainability: Propose a sustainable funding model for the new forked project to prevent future maintainer burnout.

4.

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