Open Source Software Paradigms

Lecture - 12

Open source Software Development

Lecturer: Biniam Behailu
Addis Ababa Science and Technology University

Learning Objectives

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

- ☞ Compare and contrast the Cathedral and Bazaar models of software development.
- ☞ Identify the pathways and platforms for finding open-source projects to contribute to.
- ☞ Execute the standard Git-based workflow for contributing to an open-source project.
- ☞ Apply best practices for communication and collaboration within an open-source community.
- ☞ Develop a personal strategy for growing as a sustainable open-source contributor.

Contents

Software Development Models (The Cathedral and The Bazaar)

The Cathedral and The Bazaar

- Software projects have developed using different models to control the outcome.
- As software began to be developed within a world-wide culture, particularly when the source code was freely available, **two** distinct models of development have become popular:

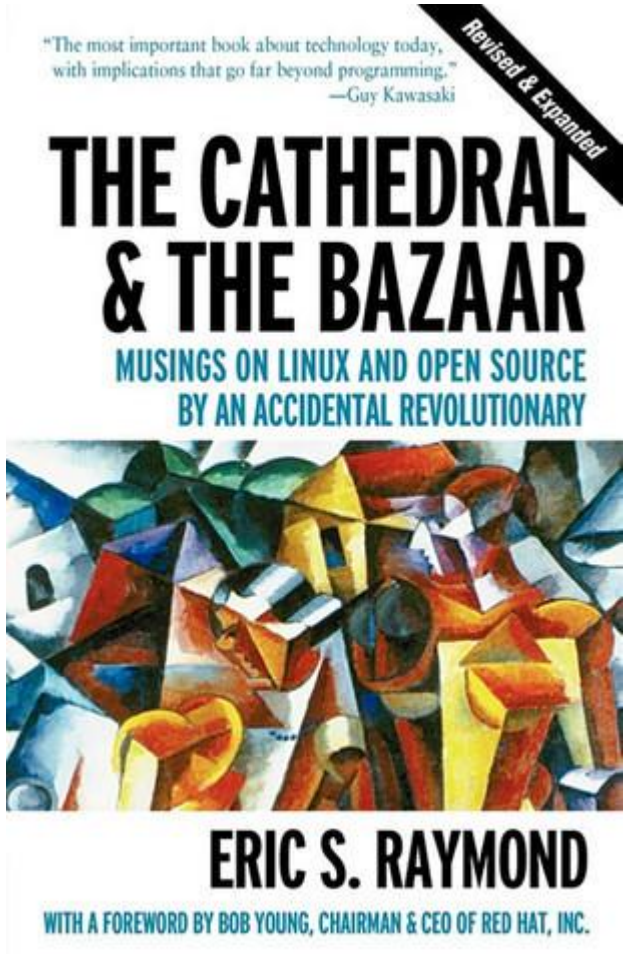
The Cathedral model

Where one or two "master builders" define the basic structure of what will be built. BSD UNIX and the GNU project are typical.

The Bazaar model

Is free running; there is no central control or basic building plan. Different programmers, like vendors at a bazaar, offer up different approaches to different problems. This approach has been the core of Linux development.

The Cathedral and Bazaar Models



- Eric S Raymond's famous 1997 essay on the Linux style of open source is here [1].
- In the book Raymond compares the Linux approach (the bazaar) to the GNU approach (the cathedral), and finds the former to be much more responsive to user needs. The essay eventually became a full-length book.

Click here to download

***Image source**: E. S. Raymond, *The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, 1st ed. O'Reilly Media, 1999. [Online]. Available: https://upload.wikimedia.org/wikipedia/en/c/c4/Cathedral-and-the-Bazaar-book-cover.jpg. [Accessed: 10-Nov-2025].*

The Cathedral Model

- Involve centralized planning with less frequent updates controlled by a small team.
- Focuses on control, structure, and thorough testing, often leading to a slower development cycle.
- Source code is released with each software release but code developed between releases is restricted. Example GNU Emacs, GCC.
- Its key aspects include
  - ☞ Centralized Development
  - ☞ Structured Process
  - ☞ Infrequent Releases
  - ☞ Limited Community Input
  - ☞ Quality Assurance

The Bazaar Model

- The "bazaar model" refers to a decentralized and collaborative approach to software development.
- Characterized by open collaboration, rapid prototyping, and community-driven development.
- Linux kernel and many open-source projects exemplify the bazaar model in action.
- Its key aspects include
  - ☞ Open Collaboration
  - ☞ Rapid Prototyping
  - ☞ Transparency
  - ☞ Diverse Contributions
  - ☞ User-Centric

The Cathedral vs Bazaar Models

- Eric S. Raymond’s framework for development styles

Feature	Cathedral	Bazaar
Development	Centralized, private	Public, open
Releases	Rare, large	Frequent, small
Governance	Top-down	Community-driven
Example	MS Office	Linux, Python

Finding Projects to Contribute To

Where to Find Open Source Projects?

- **Identify Interests -** Choose a project that aligns with your interests or skills whether it's coding documentation design or testing.
- Explore Platforms Use platforms
  - ☞ **GitHub Explore**: github.com/explore
  - ☞ **Good First Issues**: goodfirstissues.com
  - ☞ **Code Triage**: codetriage.com
  - ☞ **First Timers Only**: firsttimersonly.com
  - ☞ **Up For Grabs**: up-for-grabs.net
  - ☞ **Open Source Guides**: opensource.guide

How to Choose a Good Project?

Checklist ✅

- ☑ Recent commits (within 30 days)
- ☑ Active community and maintainers
- ☑ CONTRIBUTING.md present
- ☑ Beginner-friendly issues: Look for labels like 'good first issue,' 'beginner-friendly,' or 'help wanted.
- ☑ Clear Code of Conduct

How to Choose a Good Project?

Key files

- ☞ `README.md` — What the project does
- ☞ `LICENSE` — Legal permissions
- ☞ `CONTRIBUTING.md` — How to contribute
- ☞ `CODE_OF_CONDUCT.md` — Behavior expectations

Activity

Go to: github.com/facebook/react

Find and review:

- ☞ README
- ☞ CONTRIBUTING
- ☞ LICENSE
- ☞ CODE_OF_CONDUCT

💬 Discuss findings in

The Contribution Workflow

Overview of the Git Workflow

1. **Fork**
2. **Clone**
3. **Create a branch**
4. **Commit changes**
5. **Push to your fork**
6. **Open Pull Request**
7. **Get reviewed and merged**

Step 1 – Fork the Repo

- On GitHub → Click Fork (top-right)
- ✅ Creates your copy under your username

Step 2 – Clone Your Fork

```
git clone https://github.com/YOUR-USERNAME/project.git

cd project
```

Step 3 – Add Upstream Remote

```
git remote add upstream https://github.com/ORIGINAL-OWNER/project.git

git fetch upstream
```

Step 4 – Create a Feature Branch

```
git checkout -b feature/update-docs
```

Step 5 – Make and Commit Changes

```
git add .

git commit -m "docs: improved installation instructions"
```

Step 6 – Push Changes

```
git push origin feature/update-docs
```

Step 7 – Open Pull Request (PR)

- On GitHub → Click Compare & Pull Request
- Write PR message:

```
## Description

Improved installation steps for Windows users.


## Checklist

- [x] Tested instructions

- [x] Updated README

Fixes #123
```

Step 8 – Peer Review Process

- Reviewers leave comments
- You respond and make requested changes
- Reviewer approves → Maintainer merges

Be polite, ask clarifying questions, learn from feedback

Step 9 – Merge and Celebrate

🎉 Congratulations — your code is now part of the project!

- Acknowledge maintainers
- Share your contribution link
- Add it to your portfolio

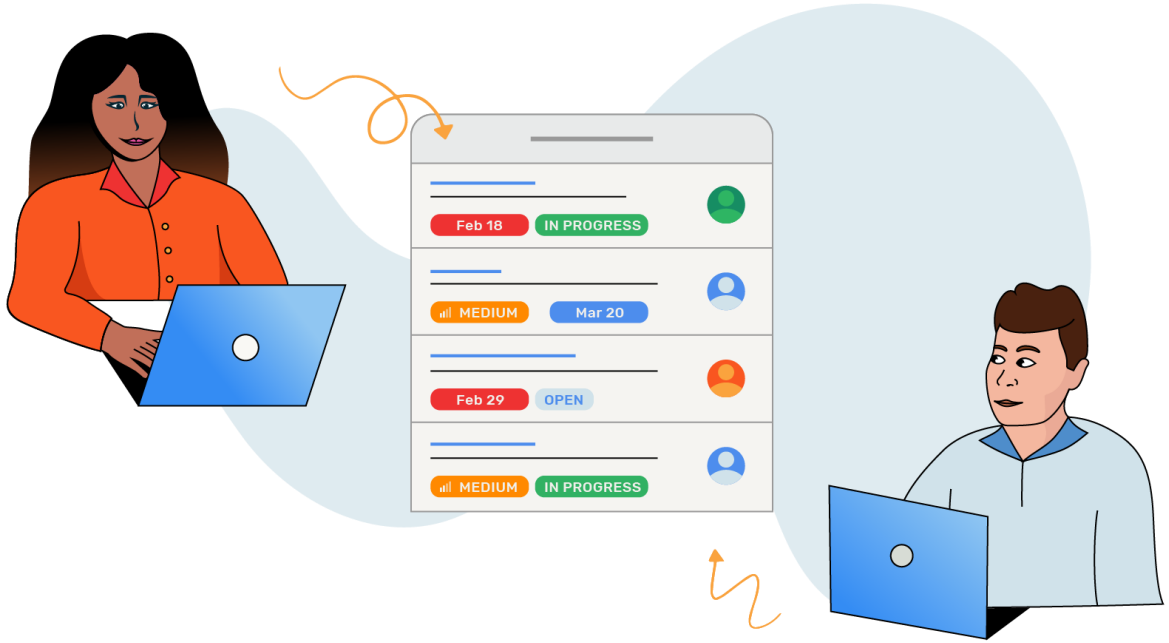
Tools and Best Practices

Common Git Commands Reference

Task	Command
Check status	git status
Stage files	git add .
Commit	git commit -m "message"
Pull updates	git pull upstream main
Push changes	git push origin branch

Issue Tracking Systems

- **Purpose**
  - ☞ Manage bugs & features
  - ☞ Assign tasks
  - ☞ Track progress



- **Tools**
  - ☞ GitHub Issues
  - ☞ Jira (for larger OSS)

***Image source**: "Why You Need an Issue Management System," [Online Image]. Available: https://cdn.prod.website-files.com/60395b77a0a8f978f0a3d9e7/6217c97a8afea7f9d6073e81_1_Why%20you%20need%20an%20issue%20management%20system.png. [Accessed: Oct. 10, 2023].*

Writing a Good Issue Report

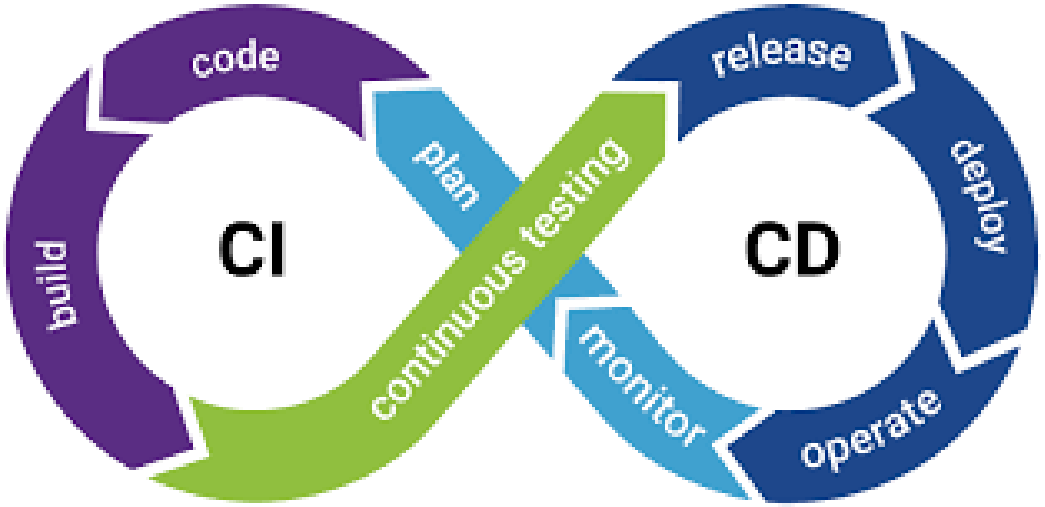
```
### Description
Login fails on mobile.

### Steps to Reproduce
1. Open site
2. Click Login

### Expected
Opens modal
### Actual
No response
```

Continuous Integration (CI/CD)

- Automation ensures quality:
  - ☞ Tests run on every PR
  - ☞ Linting & formatting enforced
  - ☞ Build must pass before merging



- Tools: GitHub Actions, Jenkins, Travis CI

***Image source**: "Continuous Integration and Continuous Deployment," [Online Image]. Available: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRAABUoTg0hRIRysVXsNZg21ojLCOSsljUEIA&s. [Accessed: Oct. 10, 2023].*

Code Review Etiquette

Reviewer

- Be constructive and specific
- Focus on code, not person

Author

- Be open to feedback
- Address all comments
- Ask questions when unclear

Example – Real Pull Request Analysis

Open: github.com/microsoft/vscode/pulls

Discuss:

- ☞ What was changed?
- ☞ Review comments?
- ☞ How was it improved before merge?

Growing as an OSS Contributor

Types of Contributions

- Code improvements
- Documentation updates
- Testing and QA
- Translation and accessibility
- Community support

Activity – FreeCodeCamp

1. Fork repo - https://github.com/freeCodeCamp/freeCodeCamp
2. Fix documentation typo
3. Open PR
4. Get review
5. Merge successfully

💡 Thousands started here!

Building Your GitHub Portfolio

Include

- ☞ Bio and pinned projects
- ☞ Clear contribution history
- ☞ README profile

Example

```
## Hi, I'm [Your Name]

💻 Contributor: freeCodeCamp, Mozilla

🌱 Learning DevOps & OSS Collaboration
```

Networking in OSS

- Join project Discord/Slack
- Attend open source hackathons
- Follow maintainers on Twitter/LinkedIn
- Contribute to discussions

Avoiding Common Pitfalls

❌ Large unreviewed PRs

❌ Poor communication

❌ Ignoring feedback

✅ Start small and consistent

Handling Merge Conflicts

```
git fetch upstream

git rebase upstream/main

# Resolve conflicts manually

git add .

git rebase --continue
```

Sustainability and Burnout

- OSS maintainers are volunteers who dedicate their personal time and effort to manage projects.
- It's important to respect their time and understand they may have other responsibilities.
- Clear and concise communication fosters positive interactions with maintainers.
- Contribute to OSS projects regularly but sustainably to avoid overwhelming them.
- Quality contributions, no matter how small, are valuable and help maintain project health.

30-Day Open Source Challenge

Week	Goal
1	Find 2 OSS projects
2	File a bug or doc update
3	Submit your first PR
4	Review someone else’s PR

Summary

- **Two Development Models**: The controlled Cathedral vs. the collaborative Bazaar.
- **Everyone Can Contribute**: Value is added through code, docs, design, testing, and support.
- **The Contribution Pathway**: A clear workflow of Fork, Clone, Branch, Commit, Push, and Pull Request.
- **Community is Key**: Success hinges on respectful communication, constructive reviews, and inclusive communities.
- **A Journey of Growth**: Open source is a pathway for learning, building a portfolio, and professional networking.

Brain Teaser

1. What is the primary characteristic of the "Cathedral" model of software development?

A. Rapid prototyping and frequent releases

B. Decentralized control and open collaboration

C. Centralized planning and infrequent releases

D. Community-driven governance

Brain Teaser

1. What is the primary characteristic of the "Cathedral" model of software development?

A. Rapid prototyping and frequent releases

B. Decentralized control and open collaboration

C. Centralized planning and infrequent releases

D. Community-driven governance

Brain Teaser

2. What is the correct sequence in the standard Git contribution workflow?

A. Clone -> Fork -> Commit -> Pull Request

B. Fork -> Clone -> Branch -> Commit -> Pull Request

C. Branch -> Commit -> Fork -> Clone -> Pull Request

D. Clone -> Commit -> Branch -> Fork -> Pull Request

Brain Teaser

2. What is the correct sequence in the standard Git contribution workflow?

A. Clone -> Fork -> Commit -> Pull Request

B. Fork -> Clone -> Branch -> Commit -> Pull Request

C. Branch -> Commit -> Fork -> Clone -> Pull Request

D. Clone -> Commit -> Branch -> Fork -> Pull Request

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

References

[1] E. S. Raymond, The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol, CA, USA: O'Reilly Media, 1999.