

What it means to develop open source software

Ross Gardler
OSS Watch

ross.gardler@oucs.ox.ac.uk
<http://www.oss-watch.ac.uk>

Overview

- What is open source, who is OSS Watch and who is Ross Gardler?
- What tools are needed to develop open source software?
- Why do we need these tools?
 - Project Management
 - Community development
- Who're you gonna call...

What is OSS Watch?

- *“promote awareness and understanding of the legal, social, technical and economic issues that arise when educational institutions engage with free and open source software”*
- Briefing notes
- Consultations
- Workshops and conferences

Open source credentials of Ross Gardler

- Committer on a number of open source projects
 - Member of Project Management Committee of some
- Member of The Apache Software Foundation
 - “Member” is a privileged position awarded in recognition of actions within the ASF
- Administrator and Mentor within the Google Summer of Code programme

What is open source?

info@oss-watch.ac.uk

What is open source?

- Software released under an Open Source Initiative (OSI) certified license
 - **Free Redistribution**
 - **Source Code**
 - **Derived Works**
 - Integrity of The Author's Source Code
 - No Discrimination Against Persons or Groups
 - No Discrimination Against Fields of Endeavor
 - Distribution of License
 - License Must Not Be Specific to a Product
 - License Must Not Restrict Other Software
 - License Must Be Technology-Neutral

Licences, licences everywhere...

- There are over 50 OSI certified licences
- 9 are regarded as widely used
- Choosing which one is right for you is *hard*
- Not for today
- “Who're you gonna call... OSS Watch”
 - info@oss-watch.ac.uk

What else can open source be?

- Usually
 - A development methodology
 - A community
- Sometimes
 - Open content
 - Free Software
 - as in free speech, not free beer
 - Freedom of the *user* as opposed to freedom of the *developer*

Open source development

info@oss-watch.ac.uk

Open source development

- Tools for all OSS projects similar
 - but the projects are different
 - Objectives
 - Team structure
 - Sustainability model
 - Governance model
- No “one size fits all” process for development
- “Who're you gonna call... OSS Watch”
 - info@oss-watch.ac.uk
 - Workshop in June
 - Individual consultations (free – thanks to JISC)

Open source development tools

- Version control
- Testing frameworks
- Issue tracking
- Communication and Collaboration
- Configuration Management
- Compilation
- Debugging
- Integrated Development Environment
- Installer
- Document Management
-?????

Why so many tools?

To make it *easy*

- Easy to build and test sources
- Easy to track
 - Project status
 - Latest updates
- Easy to contribute
 - Install/upgrade
 - Provide feedback (bug reports and feature requests)
 - Get/provide support
 - Development environment configuration

Why do we need these tools?

First set is for management

- Project Management
- Requirements planning
- IPR registry
- User/Developer Support
- Governance

Revision Control Systems (RCS)

- A developer collaboration tool
 - Central repository
 - Distributed developers
 - Revision conflict management
 - Revision notifications
 - Enables “commit-the-review”
 - Project memory
- IPR registry

Issue Trackers

- A user/developer communication tool
 - Bug tracking
 - Feature requests
- A management tool
 - Roadmap definition
 - Patch queue management
 - Enables “review-then-commit”
 - Task management

Build Tools

- Automate building
 - Faster development cycles
- Automate testing
 - Quality Assurance
- Facilitate contribution
 - Development environment independent
 - Running start
 - Tested contributions

Community development tools

info@oss-watch.ac.uk

Why do we need these tools?

Second set is for Community

- Community is key to ***sustainability*** in open source
- Build community infrastructure early
- You need a community of users
 - Requirements
 - Testing
 - Feedback
- Most projects have a ***closed*** community of developers and ***fail*** to cultivate their users

Why bother with community tools?

- **The** key to sustainable open source software
- You are the the initial community
 - Developers
 - Easy to configure development environment
 - Easy to submit patches
 - Easy to contribute to design
 - Easy to get **and** give support
 - Users
 - Easy to install/upgrade
 - Easy to make bug fixes/feature requests
 - Easy to get **and** give support
 - Observers
 - Potential collaborators/developers

Communication is the key: Discuss

- Mailing lists/Web Forums
 - Announcements – low volume, read only
 - Commits – automated notifications of all commits
 - Development – discussion of the design and development
 - Users – (not straight away) discussion of how to use the project
- Public archives are essential
- Do not make decisions off list
 - All decisions must be reported to the development

Communication is the key :

Web

- Web sites
 - What *does* the software do (features list)
 - What *will* the software do (roadmap)
 - What *changed* in the software (change log)
 - How to participate (governance model)
 - Support (documentation)
 - Downloads (release early, release often)
- Wikis
 - Encourage user feedback/documentation
 - Beware: can split communications channels

Communication is the key: News

- Web sites require a visit from the reader (Pull)
- Rich Site Summary/Really Simple Syndication (RSS)
 - News (“released X.Y.Z”, “milestone 2 passed” etc.)
 - Changes (“added feature A”, “fixed bug Y”)
- Embed RSS in your website to avoid duplication

Documentation

- First point of contribution for new users
- Make it easy for contributors
 - Mailing list with archives (or a forum if you must)
 - Wiki?
 - Can become a rabbit warren of dead ends
 - Content Management System
 - More control and features than a wiki, is it too much?
 - Public blog?
 - “name in lights”, less permanent than a wiki (no dead ends?)
 - Blog “planet”
 - People more likely to post in own blogs but little

Syndication + Documentation = Description of a Project (DOAP)

- RDF/XML schema for sharing project details
 - What the project is for
 - Latest releases
 - Resource locations
 - Mailing lists
 - News feeds
 - Downloads
 - People
 - Project Management Committee
 - Committers
- *Watch you inbox for news on this!*

Download and install

- Open source must be available to users
 - Regular binary releases
 - Regular source releases
 - Nightly builds
 - Public read access to revision control system
- Clear and concise install instructions
 - Pre-requisites
 - Configuration steps
 - Seeing immediate results

Release Early, release often

- Pros
 - Encourage user feedback
 - Encourage developer uptake
 - Demonstrate progress
 - Attract interest
- Cons
 - Requires release management
 - Management of user expectations

Manage user expectations

- Release early, release often = incomplete code
 - Includes bugs
 - Excludes features
 - **NOTE:** this is true of closed as well as open source software, the difference is open source admits it
- Be clear about status:
 - What works and what does not
 - Changes since last release
 - Purpose of release

Wow, that's a lot of tools!

- Don't worry, help is out there
- No need to self host the tools
 - What about branding? That's what your website is for
- Sourceforge, EduForge, CCPForge, Google Code
- Getting started
 - “who're you gonna call... OSS Watch”

HELP!

info@oss-watch.ac.uk

To community and beyond...

- Tools help you to follow a process
- What is the process?
 1. Welcome your users (releases, docs, mailing lists)
 2. Coach your users (docs, support)
 3. Encourage feedback (issue tracker, user support)
 4. Apply ~~user~~ contributor patches (issue tracker, RCS)
 5. Reward your contributors (governance model)
 6. Grow your community (governance model)
 7. Reach sustainability (community)

Open source development

- One size does not fit all
- “who're you gonna call... OSS Watch”
 - Workshop in June
 - Individual consultations (free – thanks to JISC)

info@oss-watch.ac.uk