

FOSS Development

Sunil Mohan Adapa
sunilmohan at fsf dot org dot in



Contents

- What is FOSS?
- Need to Understand FOSS Development
- FOSS Development Life Cycle
- Various Aspects of FOSS Development
- Tools
- Exercises
- Further Reading



What is FOSS?

- Free Software
 0. Freedom to run the program for any purpose
 1. Freedom to study and modify the program
 2. Freedom to make copies and distribute
 3. Freedom to improve the software and release
- Open Source Software as defined by OSI



Need to Understand

It was found that the failure rate of utilities on the commercial versions of UNIX ranged from 15-43% while the failure rate of public GNU utilities was the lowest at only 6%.

- Suited for research
- Baseline for many businesses now
- Hard to escape from :)



Need to Understand (contd.)

- Security
 - Many people review the program
 - Security not through obscurity
- Quality
 - Fewer bugs
 - Better architecture
 - High reliability
- Open standards
- Technology progress

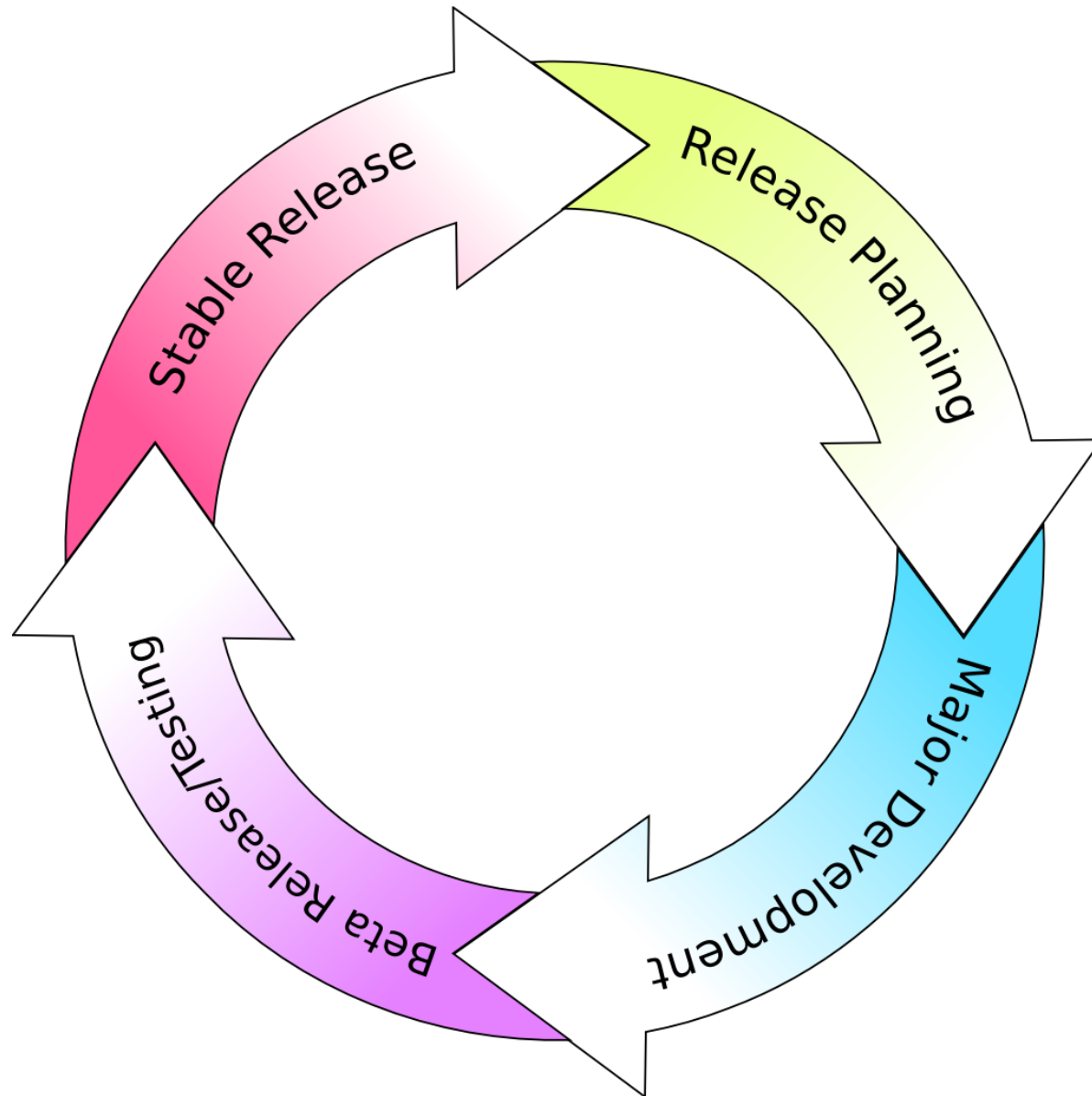


Characteristics

- Highly decentralized
- Minimal high level management
- Requirements come from users/developers
- Requires good initial architecture
- Large number of beta testers
- Very frequent releases



Typical FOSS Development Cycle



Requirements & Design

- Sometimes driven by need to replace proprietary counterparts
- Requirements are driven by developers
- Developers listen to popular demand from users



Design

- Done mostly by initial developers
- FOSS products with poor design die out
- Initial code should be clean to enable others to work on
- Premature products also get a lot of participation when idea is good
- Some projects also undergo major design changes



Code Reviews

- Depending on project policy
 - Through trust network only
 - Get multiple reviews from owners
 - Single review from committers
 - No special treatment of commit access
 - Directly commit if you have repo. access



Copyrights, Licenses, Patents and Trademarks

- Copyrights belong to the author
- Author can release under any license
- Project may look for copyright transfer
- Popular licenses (GNU GPL, GNU LGPL, BSD, MPL)
- Patents and problems
- Trademarks



Releases

- Frequent releases
- Big projects maintain stable and development branches
- Security updates
- Minor releases for bug fixing
- Automated updates
- Based on developer trust network, releases are signed



Quality Assurance

- Beta/alpha releases are very common
- Automated testing is emphasized
- Users do the major chunk of testing
- Crash reporting tools



Localisation

- Users willing to contribute
- World-wide usage and contribution
- Strong internationalisation frameworks
- Simpler tools: web interfaces, offline tools
- Sometimes direct commit access



Marketing

- FOSS spreads through word of mouth as many times they are also free of cost
- Many commercial products are now launched as FOSS products for marketing advantage
- Sometimes organized campaigns by community members



Forking

- Licenses premit complete forking of project
- A preliminary requirement to be considered FOSS
- Strongly discouraged by community
- Not very common
- Oranisationl disputes and major differences in focus lead to forks
- Friendly forks also happen



Organisation

- Mostly absent for small and medium projects
- Projects leader/maintainer mostly controls
- Foundations for bigger projects
 - Democratic process, limited power
 - Acts as guide, goal setter for the community work
 - Disposes any available funds for the set goals
- Debian social contract



Tools

- Issue tracker (bugzilla, trac)
- Communication (mailing lists, IRC, forums, offline)
- SCM (git, mercurial, SVN, CVS)
- Translations tools (pootle, kbabel)
- Documentation (wiki, web, trac, doxygen)
- Code indexing (LXR, viewvc, gitweb)
- Release (GPG, rsync, livecd)



Exercises

- 1) Pick a FOSS project that has at least moderate level of participation (> 10 contributors) and understand the processes and tools used for adding features, bug fixing, localization, submitting patches, testing, releases, branching etc.
- 2) Study the issue tracker for feature requests and bugs reports. Pick an issue however small and fix the issue by submitting a patch/solution. Understand how the fix percolates into the next release.



Further Reading

- The GNU Philosophy
<http://www.gnu.org/philosophy/philosophy.htm>
- The Cathedral & The Bazaar – Eric S. Raymond
<http://catb.org/~esr/writings/cathedral-bazaar/>



Feedback & Questions

sunilmohan at fsf dot org dot in

