

# Quality Issues in Free Software projects

Martin Michlmayr <[tbm@cyrius.com](mailto:tbm@cyrius.com)>  
University of Cambridge

This work has been funded in part by Intel.

# Objectives for today

- What is quality?
- What is quality assurance (QA)?
- Free Software and QA

# Quality

Everyone knows, but ...

- Hard to define
- Hard to measure

Definitions of quality:

- Fitness for purpose
- Attributes of quality: efficiency, reliability, usability, extendability, portability, reusability, maintainability

Different aspects:

- User perception
- Developer perception

# What is Quality Assurance?

## Traditional Quality Assurance (QA)

- Does what it should do (meets the specification)
- Does what others do as good or better as others (meets the "Industrial Standard")
- QA begins before the implementation!
  - You cannot "add" quality later
- QA is not (just) testing
- ISO defines QA as all "planned and systematic activities" (to ensure quality)

# Quality and Free Software/Open Source

Problems and challenges related to common characteristics of open source:

- Distributed development
- Done by volunteers

Naturally, different forms of the open source development model have different problems.

Eric S. Raymond (1999): The Cathedral and the Bazaar

Linus' law

# Quality and Free Software/Open Source

- Quality is often high
  - peer review (Cathedral and the Bazaar)
  - World domination
- ... but not always
  - Many small, unsuccessful projects
  - example: SourceForge has over 100,000 projects
  - Big projects have problems too
  - Contrast to QA as "planned and systematic activities"

# Interviews: identifying quality issues

Interviews with members of Free Software projects

3 main areas:

- leadership: benevolent dictator, team
  
- release cycle:
  - "release when it's ready", time based
  - fast vs slow, development vs user release
  - beta cycle, release candidates
  
- company involvement

# Interviews: underlying topics

## Processes and infrastructure

- Communication
- Bug tracking systems
- Contributing to the project

## Success

- What is success?
- Relation of success and quality?
  - Success: more volunteers
  - Contribute, Improve
  - More quality
- Cathedral to bazaar

# Techniques employed

- Infrastructure
  - Bug tracking (e.g. Bugzilla)
  - Version control (CVS, SVN, arch)
  - Automatic builds (e.g. tinderbox)
- Processes
  - Joining
  - Release: freeze, feature freeze, string freeze
  - Branches
  - Peer review
  - Testing (checklists)
- Documentation
  - Coding styles
  - Code commit

# Quality Issues

- Ports
  - keeping different ports (hardware, operating systems) in sync
  - wider issue: feature creep
- Configuration management
  - testing all possible combinations
  - (possible solution: keep it modular)
- Users don't know how to report bugs
  - many duplicates
  - bad bug reports
  - problem: long documentation -> users won't read it
- Automatic tests
  - are boring to write
  - people who write code shouldn't write tests
  - sometimes need large database
- Security updates

# Quality Issues

- Attracting volunteers
  - to triage bugs
  - to code
  - to test
  - ensure they are competent
- Lack of documentation

# Lessons learned

- Think about quality and QA
  - Debian: <http://qa.debian.org/>
  - KDE: <http://quality.kde.org/>
  - GNOME: <http://developer.gnome.org/projects/bugsquad/>
- Have developer documentation
- Have an automatic test suite
- Use version control
  - legal
- Think about quality
  - Find ways to measure quality
  - Find ways to improve quality
  - Find ways to automate quality
  - Document quality practices