

RED HAT :: NASHVILLE :: 2006

SUMMIT



Rules of Engagement, Developer edition

Benjamin Kosnik <bkoz@redhat.com>

Tom Tromeo <tromeo@redhat.com>

Introduction

- Some basics
- Specific development scenarios
 - Getting bugs fixed, pushed out
 - New features in an existing package
 - New package (ie, new development)
- Mistake patterns
- Best practice
- Thoughts on people issues
- Thoughts on support tools
- Questions and flames



Some Basics

- Have email that works
- Have internet access that is open, or access to it
- Be aware of licensing issues
- Be aware of local IP and NDA restrictions
- Familiarize yourself with project
- Ideally would have ability to test multiple sources
 - Release
 - Development



Bug fixing

- Found a bug or issue in RHEL/Fedora
- What next?
 - Isolate, test for transience or local modifications
 - Work up clear description of the issue
 - Reproduce
 - Report issue
 - Upstream
 - Elsewhere
 - Monitor feedback



Bug fixing, mistake patterns

- Fix the wrong problem
- Don't explain the scenario
- Tell a bad story
- Break the other ports
- Ignore the feedback



Bug fixing, best practice

- Report, don't diagnose
- Explain the problem
- ... and the fix



New features

- Using a package with success.
- This is almost perfect! Now I just need it to do the following....



New features, mistake patterns

- Incommunicado, then a drop: communicate with maintainers
- New feature developed on old sources
- Not 32/64 bit clean, tested on only one configuration, not tested, not documented, incorrect coding style
- Failure to get IP releases
- Underestimate time needed to integrate new work
- Failure to answer questions and keep up with sources after integration



New features, best practice

- As about new features in public place
- Wait for authoritative response
- Modify implementation plan based on feedback
- Implement
- Document
- Test, test, test
- Monitor email



New projects

- Yeah, that's cool, but not what I need



New projects, mistake patterns

- Does the world need “yet another” project? Look at existing projects and spend some time examining similar efforts. What is the real difference?
- Don't underestimate time to completion, or resources needed for maintenance.
- License ideology
- No dumping. Takes years for communities to develop.
- Living on the bleeding edge. But x/y/z needed for this, plus need ...



New projects, best practice

- Fill a new niche
- Provide support
- Provide PR
- Be patient



Thoughts on people

- Respect others
 - Some people will be difficult
 - Usually other people notice the difficult ones too
 - Maximize time spent with people doing work
- Know yourself:
 - Living on-line for years in the public: up for it?



Thoughts on tools

- Figure out email for large volume lists
 - Do you need a new email address?
 - Basic netiquette (no top posting, reply-to, etc.)
 - Figure out how to filter, spam-remove
 - Learn thread patterns
- Use mailing list archives and search tools
- Customize bugzilla
- Stay current



Additional Reading

- Social interactions
 - David Miller's "Linux Kernel Developer Social Interactions" at LCA 2006
 - <http://catb.org/~esr/faqs/smart-questions.html>
- Bugzilla
 - <http://kernelSlacker.livejournal.com/37851.html>
- Email
 - http://kitenet.net/~joey/blog/entry/thread_patterns.html

