Balancing Open Source and Corporate Objectives

James Ketrenos

July 25, 2006

SSG Core Software Division

The Dilemma

- **Meeting the needs and desires of the open source community while also meeting traditional proprietary goals is difficult**
- **Corporate methodologies and culture frequently contradict open source methodologies:**

As Greg Kroah-Hartman quoted during the keynote at the 2006 Ottawa Linux Symposium:

"Open Source development violates almost all known management theories." - Dr. Marietta Baba, Dean of the Department of Social Science at Michigan State University

Two Seemingly Different Objectives

The industry's desire to meet two different sets of objectives:

- **Corporation: provide competitive advantage, influence the industry, meet direct customer requirements (typically OEMs), control of quality, ability to create new proprietary products**
- **Community: enable freedom to innovate, develop new (and improve existing) functionality, fix bugs, maintain kernel and library compatibility, support end users, perform security audits**

Has led to two different development approaches…

Development Approach: internal only

Followed by corporations:

- **Source code and/or binaries released only after being fully developed and validated internally**
- **Support handled through corporate hosted forums or web form feedback**
- **Source changes and fixes available after being fully tested internally (frequently with several weeks or months in between releases)**
- **Bug databases are internal only**
- **Entire development, validation, and support burden carried by vendor.**
- **Perception:**

"Supporting Linux while ignoring the community."

Development Approach: 100% open

Followed by the community:

- **Source code managed fully in the open using open SCMs like git, mercurial, cvs, etc. hosted on publicly accessible servers**
- **Support handled through community forums, mailing lists, IRC, etc.**
- **Changes and fixes available immediately on mailing lists and in source repositories**
- **External project page (for example SourceForge) as the initial site users go to for information and support.**
- **Perception:**

"Supporting Linux while being a member of the community."

The Solution - Compromise

- **Enable the community to do as much as possible**
- **Only keep internal those things that the community can not contribute to**
  - Example: Certification testing
- **If you need to keep IP closed source (for example some whiz-bang algorithm), document the hardware sufficiently that the community can provide their own.**
- **Treat the community as if they were a member of your internal team**
  - Listen, and respond to, their input and feedback!!!

If you treat your beta-testers as if they're your most valuable resource, they will respond by becoming your most valuable resource. - The Cathedral and the Bazaar

Case Study: ipw3945

- **What is internal?**
  - Product road map planning
  - Execution of a full validation pass for “stable” drivers
  - Regulatory daemon development (runs on host in user space)
  - Microcode development (runs on NIC)
  - Regulatory certification testing
  - Platform bugs (if they pertain to internal details on the above)

Case Study: ipw3945 (cont.)

- **What is external?**
  - Source repository for GPL driver (using git)
    - Changes pushed to GIT as merged
    - Tip may or may not be functional; it is intended to at least always build
    - Intended to apply to latest kernel release
    - http://bughost.org/repos/?p=ipw3945.git
  - 'unstable' snapshots (using SourceForge)
    - Untested beyond build and basic functionality tests
    - Includes backward compatibility patches to 2.6.9
    - http://ipw3945.sf.net/#downloads
  - Community validation (using testrunner)
    - http://ipw3945.sf.net/#validation
  - Bug database (using bugzilla)
    - http://bughost.org
  - Support via mailing lists and IRC
    - http://lists.sf.net/mailman/listinfo/ipw3945-devel
    - irc://irc.freenode.net/#ipw2100

Case Study: ipw3945 (cont.)

- **In-tree vs. Out-of-tree**
  - "In-Tree" refers to drivers and source files that have been accepted into the mainline kernel tree managed by the Linux kernel maintainers.
  - "Out-of-tree" refers to drivers and source files that are managed outside in their own source tree.
  - The ipw3945 is currently an 'out-of-tree' driver.
    - Our out-of-tree driver provides backward compatibility for running on kernels as old as 2.6.9. If in-tree, we would only work on the latest kernel tip.
    - We are porting the driver to the latest d80211 subsystem before we want to post the driver for kernel inclusion.

Case Study: ipw3945 (cont.)

- **Set expectations of quality for external releases**
  - Stable releases have gone through a defined test pass
    - Our wireless drivers run through a combination of internal and external test cases before we label them as 'stable'
  - Unstable releases may or may not work
    - Compile smoke testing is a good idea, but with the myriad of system configurations available, even exhaustive build testing is difficult

Case Study: ipw3945 (cont.)

- **Set expectation for support to be primarily community driven**
  - Mailing lists, IRC, email
  - Engineers work on the support channels
    - You learn how the users are **really** using the product
    - You can cut through the game of telephone when trying to isolate and resolve a problem
  - Create a distinction between “enabled” features vs. “supported” features.
    - Engineers might work on “enabled” features when they have time.
    - Only “supported” features have an implied level of verified quality.

Case Study: ipw3945 (cont.)

- **Encourage community participation**
  - Explicitly note contributions in change log and snapshot release announcements.
  - Not everyone contributes by developing – a huge community contribution for the ipw3945 has been in the form of a few users taking to answering "newbie" questions, without replying RTFM :)
  - Respond to requests for features, issues, and bugs
    - If emailed directly, respond directly. If emailed on list, respond on list.

ipw3945's sister mailing list (launched in March 2004) ipw2100-devel has over eight hundred subscribers – users willing to try new snapshots and patches as soon as they are available.

The ipw2100 IRC channel has had an international community presence 24 hours a day, 7 days a week since March of 2004.

Case Study: ipw3945 (cont.)

- **Benefits realized by using this model:**
  - Enable emerging functionality within driver before it has been around long enough for it to be a marketing requirement buzzword
    - Monitor mode
    - Rtap_iface (associated monitor mode)
  - Testing on hundreds of systems in myriad of environments within minutes of releasing a new development unstable snapshot
    - Immediate feedback on whether a patch actually helped
  - Internal forces are comfortable in the validation and quality of the product
    - Anything that isn't felt to be receiving sufficient coverage by the community is explicitly tested internally.
  - Community feature contributions:
    - WPA and WPA2
    - Kernel compatibility patches
    - Improved link level quality calculations
    - Network Manager compatibility
  - Reduced cost at the same time as reduced time to market.

Summary

- **Corporation and the community traditionally use two different development approaches**
- **You can meet corporate objectives while also supporting open source methodologies by creating a compromise – enable the community to contribute wherever possible**
- **The more you enable the community, the more the community can do for you**
- **It doesn't have to be all or nothing.**

- **Recommended reading:**

"The Cathedral and the Bazaar" by Eric S. Raymond
http://www.firstmonday.org/issues/issue3_3/raymond/

Questions?


intel®
Leap ahead™