Participating in Open Source

Tim Bird
Chair, Architecture Group
of the CE Linux Forum
Overview

• Goal is to encourage you to participate in Open Source
• How to participate
  • Procedures and behaviours
• Why participate
  • Reasons to participate
“Community Effect”

• The “Community Effect” is the result from having multiple people evaluate and use the code.

• “Community Effect” is main reason for success of open source.

• Almost all guidelines for participating are meant to make it easier for other developers to evaluate and use your code.
Guidelines for Participation

• Two major categories of guidelines:
  – Mechanics – style, formatting, transmission, etc.
  – Persuasion – convincing others of the value of your code
Mechanics of Participation

- Keep current
- Make sure code style is good
- Use correct patch format
- Submit to correct place
- Use correct e-mail format
- Resources for formatting and conventions
Keeping Up To Date

• Keeping current is critical
  – If patches are not current, number of people who will use them is very small
  – Result is NO community effect
  – Kernel moves fastest – is project with biggest problem

• Is difficult challenge for embedded developers
  – CE product teams tend to “freeze” their kernel version
  – Many developers are still using 2.4.xx kernels

• Common solution is for a platform team to keep current, while product teams stay with older kernel through their development cycle
Coding Style

• See *Documentation/CodingStyle*

• Overview
  – Brace placement
  – Whitespace (use tabs = 8 chars)
    • But, lines must be less than 80 chars!
  – Variable and function naming
  – Use of goto for error handling
  – Comment style
  – Avoidance of conditional (#ifdefs)

• Best thing is to look at existing code!
Brace Placement

```plaintext
if (conditional) {
    statements;
} else {
    statements;
}

<function qualifiers> function(args)
{
    statements;
}
```
Variable and Function Names

• Examples of bad variable names:
  – ThisVariableNameIsReallyTooLong
  – tshrt
• Use underscores to separate words
  – e.g. disk_count
• Should be descriptive:
  – “foo” is bad
  – “i” and “tmp” are acceptable with limited scope
• Variables are nouns, function names usually have a verb phrase:
  – disk_count and get_free_space()
• No Hungarian notation
Use “goto” for Error Handling

```c
function () {
    item1 = allocate();
    if (error) {
        goto error_out;
    }
    item2 = allocate();
    if (error) {
        goto free1_error_out;
    }
    item3 = allocate();
    if (error) {
        goto free2_error_out;
    }
    /* more stuff */

    free2_error_out:
        free(item2);
    free1_error_out:
        free(item1);
    error_out:
        return error;
}
```
Correct Patch Format

• Use “diff –pruN linux-2.x.x.orig linux-new”
• Make patch from one level above kernel source directory
• Make patch between full original tree and modified tree
• Make sure to omit generated files
  – Use “-X dontdiff”, or better yet, put your build files somewhere outside source tree (use KBUILD_OUTPUT)
Submit To Correct Place

• Check MAINTAINERS file
• For CPU architecture sites, see:
  – http://tree.celinuxforum.org/CelfPubWiki/LinuxKernelResources
• If in doubt, ask where to submit on Linux Kernel Mailing List (LKML)
  – linux-kernel@vger.kernel.org
• If no one knows, submit patch to LKML
Use Correct E-mail Style

• Subject line:
  – [PATCH x/y] <one-line summary>

• Description
  – Include explanation of reason for change!

• Patch itself, inline in message body
  – Don’t use attachment
  – Make sure your mailer doesn’t word-wrap
  – Kernel developers want to be able to quote plaintext
Resources For Help with Mechanics

- The CE Linux Forum Patch Howto
  - http://tree.celinuxforum.org/CelfPubWiki/PatchSubmissionHowto

- Kernel source tree docs:
  - Documentation/CodingStyle
  - Documentation/SubmittingPatches.

- Andrew Morton’s “perfect patch” guidelines:

- Jeff Garzik’s guidelines:
  - http://linux.yyz.us/patch-format.html
Persuading Other Developers

• Start with explanation of problem
• Make sure code is as general as possible
• Listen carefully to criticism
  – Try to resolve issues as quickly as possible
• BE PERSISTENT!!!!
Explain Your Problem

- Explain your requirement (power management, bootup time, size, etc.)
- Other developers need to understand WHY you want this change
- Sometimes, other developers will suggest a better solution to your problem
- Even if your patch is not accepted, you have communicated a need (which might get solved in the future)
Generalizing the code

• Try to make feature useful for:
  – Other CPU architectures
  – Desktop and server developers

• This can be costly
  – Don’t overdo it
  – It is better to submit a specific patch, and get feedback, then to wait forever

• “Perfect” is the enemy of “good enough”
Persistence

• Submit your change
• If no answer, submit it again
• Even experienced kernel developers must submit multiple times
• Listen to feedback
  – Try to understand and address all feedback
  – Sometimes, feedback is very terse (brief)
Miscellaneous Advice

• Start small
  – There’s a lot of process here
  – Try to separate learning the process from dealing with technical issues
  – Start with a small bugfix

• Submit early
  – Avoid investing heavily in code that may not be accepted into mainline
Reasons to Participate

• Strategic – Long term industry benefits
  – Build ecosystem
  – Increase size of commoditized portion of software stack in products
    • Decrease overall software cost in product
    – Increased control of software in products
• Tactical – Short term, direct benefits
• Moral – Give back to community
Reason to Publish

Note that the GPL License gives you a legal obligation only to publish, not to participate in the community.
Ecosystem Thought Exercise
Magic Bowl Game

- Rules:
  - Each person starts with 100,000 won
  - On each round, any person may put some money in the bowl
  - At end of round, everyone in room receives 1/3 the amount of money in the bowl
  - Person with most money at end of game wins
Strategies for Game

A – Let others give, and only take for yourself
B – Give lots and hope others give also
C – Agree on amount to give, or somehow enforce giving
Observations

• Game models “community effect”
• Fraction received is lower in embedded community, because of fragmentation
• If not enough players, and fraction is too low, it is not worth putting money in bowl
  – e.g. 2 players, and only get 1/3 of money
• Difficult to convince management of strategies B or C
Need to recognize selfish reasons to participate in Open Source
Other Reasons to Participate

• Other people will:
  – Test your code
  – Fix bugs in your code
  – Make improvements and extensions
    • Or at least suggest them
  – Maintain your code

• This directly improves your quality and decreases your costs.
Benefit Examples

• Testing and Bugfixes
  – Printk-times patch was tested by other kernel developers, with obscure SCSI options, and a bug was found
  – Testing wouldn’t have happened in forum
  – I received patch with actual fix, not just a bug report
Benefit Examples (cont.)

• Improvements
  – Preset-LPJ patch was “taken over” by another developer, who improved it for fun
  – 3 or 4 people made suggestions and improvements
    • Restructured how option was specified
    • Printed value so configuration was easier
    • Added to kernel documentation, improved configuration help text
    • Fixed race condition
Benefit Examples (cont.)

• Maintenance
  – Preset-LPJ patch modified code that had not changed in years
  – Didn’t appear to be maintenance problem
  – A few weeks after patch was accepted, the code was moved and changed by another developer
  – CELF group did not have to do anything!
Fun and skills

• Other benefits – hard to measure
  – It’s fun!!
  – It builds skills
    • Communicating interactively with very good engineers helps build your engineering skills
    • It’s good to hear what you are doing wrong, even if it is painful
Role of CE Linux Forum

- Forum exists to help build ecosystem
  - We want to create the “Magic Bowl”
  - CELF tries to build community for CE engineers
  - Try to help companies find others interested in their features
  - Try to help build bridge from members to community
  - Try to reduce fragmentation (and thus increase “community effect”) for embedded space.
Why Participate?

You will be better
Your company will benefit
The world will be a better place
Thanks