Innovation in Open Source Software Development
A Study of the Firefox Web Browser

John Noll
Computer Engineering Department,
Santa Clara University
jnoll@cse.scu.edu

April 17, 2008
1. my research
2. innovation in open source software development
3. future directions
my research

1. tools for process modeling and analysis
   ▶ PML process modeling language.
   ▶ *pml*check process analysis tools.
   ▶ *pml*view and *PEOS* process visualization tools.

2. empirical studies of software development processes
   ▶ iterative development in student projects.
   ▶ open source software development.
   ▶ agile methods in aerospace settings.

3. (software) process enactment *PEOS* environment to support process performers.
innovation in open source software development

what drives open source innovation?

- “itch scratching” (Raymond)
  - Krishnamurthy’s data support this idea in general
- asserted requirements (Scacchi; German)
- users provide ideas, motivation (Reis & de Matties Flores; Feller & Fitzgerald)
- open source follows commercial “state of the art” (Nichols & Twidale)
- conventional requirements elicitation might work better (Henderson; Trudelle; Nichols & Twidale),
some serious itches
Firefox release 2.0 study

1. identify genesis of each feature
2. classify as:
   - proposed by user;
   - asserted by developer, from personal experience;
   - asserted by developer, from knowledge of users’ needs;
   - derived from the success of an extension;
   - motivated by a competing product;
   - result of formal requirements engineering.
example: tabbed browsing

Multizilla: H.J. van Rantwijk, (April 2001), after being rejected by Mozilla developers

Native Mozilla implementation: Dave Hyatt, v. 0.97, (December 2001)

Firefox implementation: Dave Hyatt, v. 0.3 (October 2002)

Firefox release 1.0 (November 2004)

First mention (June 1999)

2000 2001 2002 2003 2004
example: tabbed browsing

Multizilla: H.J. van Rantwijk, (April 2001), after being rejected by Mozilla developers, Native Mozilla implementation: Dave Hyatt, v. 0.97, (December 2001)

Firefox implementation: Dave Hyatt, v. 0.3 (October 2002)
Firefox release 1.0 (November 2004)

First mention (June 1999)

observations:
- asserted requirement (H.J. van Rantwijk) that was rejected
- implementation as XUL extension proves usefulness
- enthusiastic discussion by users
release 2 features

2. Phishing protection (new).
5. Resumption of previous browsing session (new).
6. Web feed (RSS) preview and subscription (enhancement).
7. Spell checking (new).
8. “Live Titles” (new).
10. Update to JavaScript version 1.7 (enhancement).
11. Search engine “plugins” in Sherlock or OpenSearch format (new).
12. Updated extension mechanism (enhancement).
13. Support for SVG text (bug fix).
release 2 features: type

- Enhancement (7)
- New Feature (6)
- Bug Fix (1)
## Results

<table>
<thead>
<tr>
<th>Feature</th>
<th>Type</th>
<th>Requirement Source</th>
<th>Initial Impl.</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual Refresh</td>
<td>enh</td>
<td>discussion forum</td>
<td>core</td>
<td>asserted, from knowledge of user needs</td>
</tr>
<tr>
<td>Phishing Protection</td>
<td>new</td>
<td>Bugzilla, Internet Explorer</td>
<td>extension (Google Safe Browsing)</td>
<td>asserted, from knowledge of user needs and competition</td>
</tr>
<tr>
<td>Enhanced Search</td>
<td>new</td>
<td>PRD</td>
<td>extension (Google search toolbar)</td>
<td>derived from extension</td>
</tr>
<tr>
<td>Improved Tabbed Browsing</td>
<td>enh</td>
<td>Mozilla Wiki, Bugzilla</td>
<td>prototype</td>
<td>asserted and formally validated</td>
</tr>
<tr>
<td>Session Resume</td>
<td>new</td>
<td>extensions (SessionSaver, Tabbedbrowser Extensions), Opera</td>
<td>extensions</td>
<td>derived from extension; competition</td>
</tr>
<tr>
<td>Web Feed Preview</td>
<td>enh</td>
<td>Bugzilla</td>
<td>extensions (RSS Reader Panel, Sage)</td>
<td>asserted, from personal experience</td>
</tr>
<tr>
<td>Spell Checking</td>
<td>new</td>
<td>Bugzilla</td>
<td>extensions (SpellBound, Torisugari)</td>
<td>user contributed</td>
</tr>
</tbody>
</table>
results, continued

<table>
<thead>
<tr>
<th>Feature</th>
<th>Type</th>
<th>Requirement Source</th>
<th>Initial Impl.</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Live Titles</td>
<td>new</td>
<td>discussion forum, developer web log</td>
<td>prototype</td>
<td>asserted, from personal experience</td>
</tr>
<tr>
<td>Improved Add-ons Manager</td>
<td>enh</td>
<td>Mozilla wiki</td>
<td>core</td>
<td>asserted, from personal experience</td>
</tr>
<tr>
<td>Javascript 1.7</td>
<td>enh</td>
<td>PRD</td>
<td>core</td>
<td>asserted, from personal experience</td>
</tr>
<tr>
<td>Search Plugins</td>
<td>new</td>
<td>Safari, Mycroft, Sherlock</td>
<td>core</td>
<td>competition</td>
</tr>
<tr>
<td>Updated Extension Mechanism</td>
<td>enh</td>
<td>Bugzilla</td>
<td>core</td>
<td>user contributed</td>
</tr>
<tr>
<td>SVG Text Support</td>
<td>fix</td>
<td>Bugzilla</td>
<td>core</td>
<td>user contributed (bug)</td>
</tr>
<tr>
<td>Windows Installer</td>
<td>enh</td>
<td>Bugzilla</td>
<td>prototype</td>
<td>asserted, from knowledge of user needs</td>
</tr>
</tbody>
</table>
genesis of features

- Asserted, from own exp. (4)
- Asserted, from knowledge of users (4)
- Extension (2)
- User RFE (3)
- Competition (1)
open source is different
conclusion

- open source *is* different
- what could conventional software projects adopt?
  - open communication channels
  - extension mechanisms
  - multiple paths to release
conclusion

- open source *is* different
- what could conventional software projects adopt?
  - open communication channels
  - extension mechanisms
  - multiple paths to release
- limitations
  - not an inside view
  - single project
  - fifteen features
conclusion

- open source is different
- what could conventional software projects adopt?
  - open communication channels
  - extension mechanisms
  - multiple paths to release
- limitations
  - not an inside view
  - single project
  - fifteen features
- future
  - expand to different domains
some observations

- open source projects use relatively simple tools: CVS, email, web sites, BugZilla.
some observations

- open source projects use relatively simple tools: CVS, email, web sites, BugZilla.
- processes are complex
some observations

- open source projects use relatively simple tools: CVS, email, web sites, BugZilla.
- processes are complex
NetBeans requirements and release process (detail)

- start
  - Review NetBeans
  - Set Release Date
  - Review NetBeans Vision Statement
  - Review Uncompleted Milestones From Previous Release
  - Review Issuezilla Feature Requests
    - Compile List Of Possible Features To Include
    - Categorize Features Proposed Feature Set
    - Send Message To Community For Feedback
      - Review Feedback From Community
      - Revise Proposal Based On Feedback
        - Post Final Development Proposal To NetBeans Website
          - Assign Developers To Complete Project Milestones
  - end
observations, continued

- multicore technology = 1000-core processors?
observations, continued

- multicore technology = 1000-core processors?
observations, continued

- multicore technology = 1000-core processors?

Before: task → artifact

After: task → agents → artifact
create_working_directory

Name: create_working_directory
State: RUN
Script: Create a working directory to contain the java files that implement your tests. You must set permissions on the path to your working directory so that the PEOS web interface can traverse the path and read your test files.

% cd
% chmod a+X ,
% mkdir coen285
% chmod a+X coen285
% cd coen286
% mkdir web_test
% chmod a+Xr web_test

Please pay particular attention to the last `chmod`; you must make your working directory both executable (`+X`) and readable (`+r`), so that the PML Web interface can read and display your files when requested. The other directories need only be executable.

Note: this only grants read access to your working directory; and only allows others to traverse, but not read, the directories in the path leading to your working directory. This enables the PEOS web interface to find your test files and other resources in your working directory, but does not allow anyone to actually list any of your directories except for your working directory.
... becomes web 3.0 - “the game web” (Scacchi)
This work is supported in part by the National Science Foundation under Grant No. IIS-0205679, through subcontract from the University of California, Irvine; and by a grant from the school of engineering at Santa Clara University. No endorsement is implied.