Master Defense
Moritz Neeb
9.02.17

Tracking Innovation in Open Source Software Projects
Innovations in Open Source Software

- OSS is under a license that allows redistribution and modification
- OSS is developed in an open, “asynchronous” way by developers around the world
- Innovation Introduction in OSS is an orchestrated (i.e. active) change to the development process
Research Goal
Motivation

• The field of OSS is not explored enough compared to its relevance
• Understanding OSS developers can be extrapolated
• Knowledge about Innovation helps new OSS projects
Research Questions

• How are innovation episodes of 2007 continued during 2008-2016?
• What constraints are given for the search for these episodes?
• How can the new data be integrated into existing theory?
Do continuations of 2007 episodes exist?

- Around 130 “innovation episodes” were identified for the year 2007
Roadmap

1) Background Information
2) Search Process
3) Results on Concept Level
4) Conclusion
5) Discussion
Background
Definitions

- **Innovation**: Either the process/system/tool under discussion or the one in place.
- **Episode**: An attempt by one or more participants to change an innovation.
Success

An innovation is successfully introduced, when:

1) it is used on a routine basis and it has solved the problem it was designed to solve, or

2) it attained the goal it was designed to attain.
Introduction Lifecycle

- Discussion Barrier
- Execution Barrier
- Usage and Adoption Barrier

Discussion Phase:
- Abandoned
- Dead-ended
- Rejected
- Killed
- Postponed

Execution Phase:
- Failed

Usage Phase:
- No Adoption
- No Goal Attainment

Successful Introduction:
- Success

Unsuccessful Introduction:
- Failed
- ???

Source: Dissertation C. Özbek
Episode Topics
Data that was used

- Mailinglist data
- Annotated mails (coding)
- Second channel information
  - code repositories
  - projects website
  - wikis
The Search Process
Motivation for a search method

emails

2007

2007-2016

0

100000

200000

300000

400000

500000

600000
emails @ GRUB
Keyword Search

- Signal-to-noise ratio
- Difficulty to find innovations
- Evolution of SCM innovation
Identified continuations per innovation

- SCM – all (8)
- GSoC – all (6)
- Bug Tracking – 4/6
- License – 2 / 5
- Version Naming – 1 / 3
Signal-to-noise ratio of different innovations

SNR (qualitative)

version name

bug tracker

gpl

low

SNR

(high)

gsoc

git

bzr

svn

(gpl)
Finding continuations

- Easy to check success, e.g. via
  - “git still in use?”
  - “license still the same?”

- Hard to find unprecedented problems, e.g.
  - “problems with license/git?”

- But with evidence, hit-rate is very high, e.g.
  - “there was a chance of version number scheme since 2007 – when?”
More evidence: developer activity
## SCM switches

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ArgoUML</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bochs</td>
<td>CVS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SVN</td>
</tr>
<tr>
<td>Bugzilla</td>
<td>CVS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SVN</td>
</tr>
<tr>
<td>Flyspray</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SVN</td>
<td>git</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FreeDOS</td>
<td>CVS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SVN</td>
<td></td>
</tr>
<tr>
<td>gEDA</td>
<td>CVS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>git</td>
<td></td>
</tr>
<tr>
<td>GRUB</td>
<td>CVS</td>
<td>SVN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>git</td>
</tr>
<tr>
<td>KVMv</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>git</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MonetDB</td>
<td>CVS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mercurial</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROX</td>
<td>CVS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>git</td>
</tr>
<tr>
<td>RequestTracker</td>
<td>SVN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>git</td>
</tr>
<tr>
<td>U-Boot</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>git</td>
</tr>
<tr>
<td>Xfce</td>
<td>SVN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>git</td>
</tr>
</tbody>
</table>
git is becoming popular
The search process

Summary

- Keyword search works fine for most of the relevant cases
- Evidence of change helps a lot
  - via emails: e.g. author activity
  - second channel: e.g. project's infrastructure
- Pressure from “outside” helps a lot for innovations
Results on the conceptual level
There were two groups of innovation continuations:
- core episodes
- peripheral episodes

These groups have very different “barrier-heights”

The visibility on the mailinglist is also different
Core Episodes

• Innovation is used on a daily basis
• If there is a problem, it will be discussed (urgency high)
• high numbers of people affected
  → continuation is likely
Peripheral Episodes

- “One-shots” or minor changes
- Can still be important for progression
- Continuation is unlikely (did not find any)
- Probably hard to find on their own without knowledge of their existence
Innovation Lifecycle is a circle indeed

- Discussion Barrier
- Execution Barrier
- Usage and Adoption Barrier

Discussion
- Problem? / Idea?
  - Continuation
  - Abandoned
  - Dead-ended
  - Rejected
  - Killed
  - Postponed

Execution
- No Execution

Usage
- Introduction
- Success
- Unknown
- No Adoption
- No Goal Attainment

modified version of source: Dissertation C. Özbek
Continuation: From problem to solution

- to let this happen, an innovator has to step up again
- evidence of repeated innovation attempts
  - autotools @ Xfce
  - SCM @ GRUB
State change / perspective change: From a solution to a problem

- A successful episode can turn into a problem or a new idea to improve the situation

- evidence:
  - bug tracker @ GRUB (process is “rotten”)
  - SCM @ GRUB (data loss motivates innovators)
Results summary

• Keyword search on mailinglists reveals enough material to evaluate the situation
• SCM management systems progression
• Lifecycle revision
• “Concept” validation
Conclusion

- The high-level “innovation processes” have not changed much
- Mailinglist communication reveals the controversial topics, but “success” is difficult to see (i.e.: silence means a good thing)
- The “concepts” of 2007 are still valid, but might need a restructuring
- The “outside progression” keeps projects evolving
Thanks for your attention.
Discussion

• Firefox WebExtensions as outside pressure?
• Are peripheral episodes important?
• Relationship to Q&U research: the projects' “develop-ability”?

• Your questions?