Open source software peer review practices:

A case study of the Apache Server

Peter C. Rigby, Daniel M. German, Margaret-Anne Storey
University of Victoria, BC, Canada

Talk by:
Patrick Bänziger

Software Engineering Seminar, ETH Zürich
March 30, 2010
Most scientists regarded the new streamlined peer-review process as “quite an improvement.”
Motivation

• Formal peer review improves software quality
• But: Adoption remains low in industry

• Apache HTTP Server project practices code review
• Very successful, mature project

• Unlike industrial peer review, scientific literature at that time patchy about OSS peer review
Research goal

- Determine parameters of Apache peer review
- Compare to traditional inspections in industry
- Identify techniques – transferrable to industry?
Most used web server since 1996
Review originated in trust issues
Quality aim: “robust, commercial-grade, featureful”

Code repository: Write access limited to core group
“Apache Project is a minimum-threshold meritocracy”

Two main review techniques
Review techniques

Review-then-Commit (RTC):
- For non-core members
- Every change must be submitted to mailing-list
- Requires $\geq 3$ binding positive votes and no veto

An accepted RTC commit is denoted as $RTCA$

Commit-then-Review (CTR):
- Possible for members of core group (Apache group)
- Commit is valid unless a core group member vetoes
Methodology

Subversion

Commit logs

Mailing list trawling

Manual examination
Data Limitations

1. Not every contribution has replies (CTR) → ignoring contributions with no reply
2. Sample is not random
3. Assumption: Message equals a review
4. Some questions can’t be answered
   - What fraction of contributions is reviewed?
   - How long does a reviewer need per contribution?
Questions and Findings

Q: Are developers able to maintain adequate level of review during times of increased development activity?

- CTR: **Yes**, reviews scale proportionally ($r = 0.69$) to number of commits
- **Standards are not lowered** – accepted contributions correlate weaker with RTC commit count ($r = 0.37$)
Q: What is the artifact size? Why is it so small?

<table>
<thead>
<tr>
<th></th>
<th>RTC</th>
<th>RTCA</th>
<th>CTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>median</td>
<td>25 lines</td>
<td>17 l.</td>
<td>15 l.</td>
</tr>
<tr>
<td>80%</td>
<td>&lt; 106 lines</td>
<td>&lt; 71 l.</td>
<td>&lt; 70 l.</td>
</tr>
</tbody>
</table>

- RTCA has much smaller commit size than RTC
- Manual inspection: Larger commits often broken up
- Apache Foundation: Policy of 'one patch per issue'
Comparison RTC / CTR

- CTR is faster with first response and review duration

<table>
<thead>
<tr>
<th></th>
<th>RTC</th>
<th>CTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to first response (80%)</td>
<td>&lt; 1 d</td>
<td>&lt; 11 h</td>
</tr>
<tr>
<td>Review interval (median), if defect found</td>
<td>19 h</td>
<td>8.9 h</td>
</tr>
</tbody>
</table>

- No significant difference in number of defects found
Industry comparison:

• Lucent Technologies performed inspection experiment
• Artifacts at Lucent are larger (about 5.5x)

• Apache RTC is 56% faster than Lucent meetings (compensated for artifact size)

• Previous research showed that asynchronous reviews are approximately 25% faster
Threats to Validity / Criticism

- Only one project – due to 'space limitations'
- Data about workload on developer not available
- Data limitations:
  - Positive (silent) reviews in CTR not countable
  - Not all posts can be properly assigned to a review
- Source referencing fragmentary
- Raw data and tools not available
Conclusions

- **CTR:**
  - is faster than RTC, without loss of effectiveness
  - still works when commit numbers peak

Applications for industry:

- Small artifacts: possibly faster review
- Asynchronous reviews may improve SW quality without large burden on developers
- CTR for senior programmers, RTC for new hires
Appendix
Questions and Findings

Q2: How many reviewers respond? How much messages? What is the size of the review group?

<table>
<thead>
<tr>
<th></th>
<th>RTC</th>
<th>RTCA</th>
<th>CTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reviewers per contrib.</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Messages per review</td>
<td>2</td>
<td>N/A</td>
<td>2</td>
</tr>
<tr>
<td>Review group size (/month)</td>
<td>14</td>
<td>N/A</td>
<td>15</td>
</tr>
</tbody>
</table>

All values are medians
Defects Discovered

Q5: How many reviews did find a defect?

- Manually sampled for RTC / CTR, N=100 each:

<table>
<thead>
<tr>
<th></th>
<th>RTC</th>
<th>CTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect thereof:</td>
<td>51%</td>
<td>66%</td>
</tr>
<tr>
<td>Abstract</td>
<td>16%</td>
<td>34%</td>
</tr>
<tr>
<td>Source</td>
<td>37%</td>
<td>32%</td>
</tr>
<tr>
<td>Both</td>
<td>47%</td>
<td>30%</td>
</tr>
</tbody>
</table>
Artifact size

<table>
<thead>
<tr>
<th></th>
<th>RTC</th>
<th>RTCA</th>
<th>CTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>median</td>
<td>25 lines</td>
<td>17 l.</td>
<td>15 l.</td>
</tr>
<tr>
<td>80%</td>
<td>&lt; 106 lines</td>
<td>&lt; 71 l.</td>
<td>&lt; 70 l.</td>
</tr>
</tbody>
</table>

Manual examinations in RTC:

Large contributions often by 'neophytes' (new developers)

These were often broken up into components in review
Industrial vs. Apache review