Developer Dynamics and Syntactic Quality of Commit Messages in OSS Projects

Kuljit Kaur Chahal, Munish Saini

Department of Computer Science
Guru Nanak Dev University, Amritsar, India
kuljitchahal.cse@gndu.ac.in
Open Source Software Development

- Community plays an important role
- Volunteer participation
- Dynamic Community
  - No fixed roles
  - People can leave/join any time
    - Lean/active periods
- to investigate community dynamics and its impact on OSS development process.
  - understand the impact of community dynamics on the quality of contributions committed to a project’s repository.
Existing work related to Commit Analysis, Community Dynamics and Software Quality

- Most of the works in the research literature on commit analysis of OSS projects deal with identifying:
  - commit size distribution [3],
  - commit frequency distribution [13],
  - commit characterization [17, 23], and
  - Contributor’s commit activity distribution [8].
    - Chełkowski et al. [8] analysed commit contributions of Apache contributors to highlight inequalities among open source contributors’ in producing content in the OSS paradigm which is often described as collaborative.

- Code quality decreases as the number of contributors increases [2]
- Quality of components (measured using the number of defects) developed by distributed teams was bad in comparison to the quality of components developed by collocated teams[7].
The Research Agenda

- Lack of literature on the subject and the broad nature of practitioner recommendations
  - the Multi-vocal Literature Review (MLR) approach [8].
- Measuring commit message quality by using 11 metrics related to the syntax of a commit message.
- To explore if there is any relation between community dynamics and the commit message quality of the OSS projects.
What is a Good Commit?

- A good quality commit contains a well-crafted message with all the necessary details (meta-data) to effectively convey the change to current or future developers [5].
- A good commit message should follow a simple and consistent style for specifying commit meta-data and content.
Table 1. Rules for writing a good commit

1. Title (subject line) of commit message should be short (between 50-72 characters).
2. Subject line should end with a dot.
3. Capitalize the subject line i.e. first character of the subject line should be capital.
4. Use imperative mood in the subject line for example use words like fix, add, update in place of fixing, adding, and updating etc.
5. Subject line should be concise and limit the number of “and”, “or”.
6. Subject line should not include details such as bug number, file name, ticket number, and any other external references.
7. Subject line and body must be separated by a blank line.
8. Body of a commit message must have multiline description. It should be well explanatory detailing why and what is changed.
9. Body of a commit message should not contain lots of bullets, hyphens, or asterisks.
10. A Commit should have one logical change.
Table 2. Commit Message Syntactic Quality Measures

<table>
<thead>
<tr>
<th>Commit Quality Measures</th>
<th>Commit Score</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length of Title</td>
<td>=0 or &gt;72</td>
<td>1-10 11-30 31-50 51-72</td>
</tr>
<tr>
<td>Title ends with dots</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Title first character capital</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Count number of “and” “or” in Title</td>
<td>&gt;6</td>
<td>5-6 3-4 1-2 0</td>
</tr>
<tr>
<td>Count number of “file name” in title</td>
<td>&gt;6</td>
<td>5-6 3-4 1-2 0</td>
</tr>
<tr>
<td>Count number of external references in title</td>
<td>&gt;6</td>
<td>5-6 3-4 1-2 0</td>
</tr>
<tr>
<td>Imperative mode in title</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Commit body existence</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Count number of “file name” in body</td>
<td>0</td>
<td>10&gt; 6-10 3-5 1-2</td>
</tr>
<tr>
<td>Count number of external references in body</td>
<td>0</td>
<td>10&gt; 6-10 3-5 1-2</td>
</tr>
<tr>
<td>Count number of paragraph in body</td>
<td>0</td>
<td>10&gt; 5-10 3-4 1-2</td>
</tr>
</tbody>
</table>
Measuring the Commit Message Quality

1. Normalize the scores of individual measures to a common scale [0, 1].
2. Calculate the weighted average to find the total commit score
3. Validated the Metric definitions using survey based approach
   - 20 participants
   - 16 graduate and 4 under-graduate
   - 5 to 7 years experience in software projects based on Java/C#
   - The participants were asked to upvote a rule if they agree, downvote a rule if they don’t agree, or post a neutral response if it does not matter to them while reading a commit message.
   - to further validate the results of the commit message quality score, the results of the proposed model for a sample of 100 commits messages (50 with commit messages as per the rules and 50 otherwise) were compared with the assessment results made available by the same survey participants.
   - The results show that 84% of the commit messages were correctly judged by the proposed model. Specifically, for commit messages with good quality, about 88% of messages were correctly judged, and about 80% of messages with poor quality were correctly judged by the proposed model.
### Table 4. Descriptive Statistics of the OSS projects

<table>
<thead>
<tr>
<th>OSS Projects</th>
<th>Origin Date</th>
<th>Number of Months</th>
<th>Number of Contributors</th>
<th>Commit messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostgreSQL</td>
<td>Jul, 1996</td>
<td>239</td>
<td>43</td>
<td>54355</td>
</tr>
<tr>
<td>glibc</td>
<td>Feb, 1989</td>
<td>321</td>
<td>410</td>
<td>43313</td>
</tr>
<tr>
<td>Eclipse-CDT</td>
<td>Jun, 2002</td>
<td>168</td>
<td>203</td>
<td>28817</td>
</tr>
<tr>
<td>GnuCash</td>
<td>Nov, 1997</td>
<td>222</td>
<td>105</td>
<td>21969</td>
</tr>
<tr>
<td>WordPress</td>
<td>Apr, 2003</td>
<td>158</td>
<td>73</td>
<td>37333</td>
</tr>
<tr>
<td>Firebug</td>
<td>Aug, 2007</td>
<td>181</td>
<td>45</td>
<td>13043</td>
</tr>
<tr>
<td>Rhino</td>
<td>Apr, 1999</td>
<td>105</td>
<td>56</td>
<td>3721</td>
</tr>
</tbody>
</table>
Analyzing the quality of commit messages in the OSS projects
Fig. 1. Variation in commit quality of the OSS projects.
Does the number of contributors affect the commit message syntactic quality?
Fig. 3. Variation in the number of contributors of the OSS projects
Contributor churn of the OSS projects over the period of time
Understanding the Contribution Pattern

In this regard, the first step is to find commit distribution among different contributors of the OSS projects to identify the core group of contributors.
Next, we analyze their commit behavior from two perspectives – commitment (i.e. regularity to commit), and the level of skill (i.e. commit message quality)
<table>
<thead>
<tr>
<th>OSS Projects</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
<th>C5</th>
<th>C6</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostgreSQL</td>
<td>34.5</td>
<td>26.52</td>
<td>7.35</td>
<td>3.62</td>
<td>3</td>
<td>0</td>
<td>21.55</td>
</tr>
<tr>
<td>glibc</td>
<td>40.5</td>
<td>24.18</td>
<td>4.72</td>
<td>4.14</td>
<td>6</td>
<td>6</td>
<td>19.46</td>
</tr>
<tr>
<td>Eclipse-CDT</td>
<td>10.1</td>
<td>7.90</td>
<td>6.43</td>
<td>5.64</td>
<td>6</td>
<td>2</td>
<td>58.84</td>
</tr>
<tr>
<td>GnuCash</td>
<td>16.6</td>
<td>14.84</td>
<td>12.65</td>
<td>7.93</td>
<td>7</td>
<td>7</td>
<td>32.78</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>22.1</td>
<td>4.7</td>
<td>3.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Commit regularity
Analyzing Commit Regularity and Commit Quality
PostgreSQL
WordPress
Conclusions

- The major objective of this study was to understand the impact of community dynamics on the quality of contributions submitted to a source code management system of an OSS project.
- A commit message quality model is proposed to evaluate the syntactic quality of commit meta-data submitted by the contributors of an OSS project.
- Commit quality improves when multiple contributors become active at the same time (PostgreSQL, glibc, GnuCash).
- In some cases (Wordpress and Firebug), commit quality degrades when some contributors start contributing to the project repository.
Future Work

• To analyze the semantic quality of commits
• To analyze the commit message quality of different types of commits such as corrective v/s non-corrective
• To investigate the relevance of commit message quality with quality of the code contributed as part of commits
• To profile developers on the basis of the quality of their contributions for developer labeling.
References

Thanks