Prediction Software Quality from Development and Release Factors

Authors: Rishita Mullapudi, Tajmilur Rahman, Joshua Nwokeji

Presenter:
Tajmilur Rahman PhD
rahman007@gannon.edu
Tajmilur Rahman, PhD, is an assistant professor in the department of Computer and Information Science at Gannon University in Erie, Pennsylvania, United States. Dr. Rahman received his doctorate degree in 2018 from Concordia University, Montreal QC, Canada.

His overarching research interest is to investigate release engineering practices in software systems. His research interests also include software engineering & data science, understanding the significance of software architecture for long lasting software systems, and providing tool support to the community to nurture software feature architecture. He is also interested in studies on software engineering education.

Dr. Rahman is the first author who extracted feature-architecture. His currently ongoing research works include, software release management, software quality prediction in rapid-release, predict potential architectural drift, and software engineering education.
Outline

- Software quality
- Importance of software quality
- Quality in terms of post release bugs
- Shorter version release cycles
- Development and release factors
- Data
- Methodology
- Preliminary results
Software Quality

- Functional Quality
- Quality in Performance
- User Experience (UI) Quality
- User Experience - Post Release Issues
Software Quality - Post Release Bugs

- Software crash on the users’ end
- Bugs reported by users
  - Example: Browser crash-report
Software Quality - Post Release Bugs

- Software crash on the users’ end
- Bugs reported by users
  - Browser crash-report
  - Operating System crash report
Software Quality - Post Release Bugs

- Reports are automatically collected from the end user
Software Quality - Why Important

- Indicates the quality of the released version
- Affects uner satisfaction
- Adds work load into the future sprints
- Additional effort slows down desired pace
Shorter Version Release Cycle

- Promised delivery of new version
- Every certain period of time
- Typically days or weeks
- Limited time for testing
- May lead to more post release bugs
Shorter Version Release Cycle

- Better to predict post release bugs
- Help developers be cautious
- Help prioritizing tests
Development and Release Factors

1. Number of Commits
2. Number of Churn per File
3. Number of Churn per Test File
4. Number of Churn per Configuration File
5. Number of Last Minute Churns
6. Number of Bug-fix Commits
Development and Release Factors

Number of Commits

- Indicates the volume of work done in a release
- Indicates activeness of the developers
Development and Release Factors

Number of Churn per File

- Indicates the volume of work in each file
- Indicates cluster of development effort
  - Amount of work for feature development
  - High churn per – may be a few features received a lot of work
  - Low churn per file – may be enhancement or modifications or small feature
Development and Release Factors

Number of Churn per Test File

- Indicates the volume of testing effort
- Also indicates presence of many features
Development and Release Factors

Number of Churn per Configuration File

- Indicates the volume of configuration effort
- Multiple platform release
- Infrastructure change
Development and Release Factors

Number of Churn at the Last Moment

- Indicates rush among the developers
- Last moment ~ Last one month of changes
- If the last commit before the release commit is May 31, then changes during May 01 - May 31 are considered as last minute changes
Development and Release Factors

Bug-fix Changes

- Amount of changes to fix bugs
- Identified from the commit messages
  - Patterns used: “Bug”, “Fix”, “Patch” etc.
Data

- Bug data
  - Eclipse Equinox bugs
  - https://bugs.eclipse.org/bugs/xmlrpc.cgi

- Historical repository data
  - Github repository for Eclipse development
  - https://github.com/eclipse/rt.equinox.p2
<table>
<thead>
<tr>
<th>Version #</th>
<th>commit</th>
<th># churn</th>
<th># file</th>
<th>Conf Churn</th>
<th># LM Churn</th>
<th># TF Churn</th>
<th>m Bugs</th>
<th>M Bugs</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0</td>
<td>10k</td>
<td>20k</td>
<td>500</td>
<td>20</td>
<td>10k</td>
<td>200</td>
<td>180</td>
<td>54</td>
</tr>
<tr>
<td>3.1</td>
<td>15k</td>
<td>25k</td>
<td>400</td>
<td>12</td>
<td>12k</td>
<td>100</td>
<td>223</td>
<td>78</td>
</tr>
<tr>
<td>3.2</td>
<td>12k</td>
<td>22k</td>
<td>450</td>
<td>21</td>
<td>15k</td>
<td>300</td>
<td>124</td>
<td>98</td>
</tr>
</tbody>
</table>
Classification

5 Decision Classes:

- Quality L1
  - Quality magnitude 0 - 5
- Quality L2
  - Quality magnitude 51 - 100
- Quality L3
  - 101 - 150
- Quality L4
  - 151 - 200
- Quality L5
  - 201 +
Quality Magnitude

Indicates the magnitude of post release bugs for a release. The more high-impact bugs, the higher the magnitude is.

\[ \text{magnitude} M = \begin{cases} \text{hb} & : \text{hb} > 0 \\ \text{hb} \times (\text{mb} + \text{Mb}) \times 100 / \text{Tb} & \end{cases} \]

- hb <- High-impact bugs
  - Blocking and critical bugs
  - Will be applied only if it is > 0
- mb <- Minor bugs
- Mb <- Major bugs
- Tb <- Total bugs
Quality Magnitude

Magnitude of release version 3.4.0

Critical + Blocking bugs = 19

Minor + Major bugs = 26

Magnitude = $m_{3.4} = 19 \times (26 \times 100/644) = 76.7$

Falls under Quality Level 2 [51 - 100]
## Quality Magnitude

<table>
<thead>
<tr>
<th>Version #</th>
<th>Min</th>
<th>Maj</th>
<th>Crit</th>
<th>Block</th>
<th>Total Bugs</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.0</td>
<td>6</td>
<td>20</td>
<td>12</td>
<td>7</td>
<td>644</td>
<td>76.7</td>
</tr>
<tr>
<td>3.5.0</td>
<td>14</td>
<td>17</td>
<td>5</td>
<td>8</td>
<td>349</td>
<td>115.0</td>
</tr>
<tr>
<td>3.6.0</td>
<td>3</td>
<td>11</td>
<td>5</td>
<td>1</td>
<td>180</td>
<td>47.0</td>
</tr>
</tbody>
</table>

Quality magnitude for release version 3.4.0, 3.5.0, 3.6.0
Classification Models

- Decision Tree
- Random Forest
- Naïve Bayes
- Support Vector Machine
Method At a Glance
In Progress

● 61% of the work is done
● Will apply K-Means clustering to find the classification thresholds
● Will construct the decision tree, random forest and other classification
● Compare their performance
Thank you for Being My Audience

Questions or comments are always welcome

Tajmilur Rahman PhD
rahman007@gannon.edu