Mining Peer Code Review System for Computing Effort and Contribution Metrics for Patch Reviewers

Rahul Mishra       Ashish Sureka
{rahulm, ashish}@iiitd.ac.in

4th Workshop on Mining Unstructured Data (MUD 2014)
Peer Code Review

- Code review is *systematic examination* of computer source code
- Intended to *find and fix mistakes* overlooked in the initial development phase
- Improving both the overall *quality of software* and the developers' skills

- Code review practices fall into following main categories
  - ✓ Over-the-shoulder
  - ✓ Email pass-around
  - ✓ Pair Programming
  - ✓ Tool-assisted code review

Google Chromium – Rietveld Code Review System

Issue 451803002: V8 Buildbot: Reorganize slaves.

- Description
  V8 Buildbot: Reorganize slaves.
  This removes support for v8-naocl2. The fyi builders grouped with a distinct set of slaves.
  TBR-rietveld@chromium.org

- Patch Set 1
- Patch Set 2:

  Total comments: 1

  Created: 13 minutes ago
  Unified diffs
  M masters/master.client.v8/master.cfg
  M masters/master.client.v8/slaves.cfg

  Trybot results: Sign in to try more bots
  Commit: 

  Messages
  Total messages: 5
  Expand All Messages | Collapse All Messages
  Michael Achenbach: The CC list was checked by rietveld@...
Google Android – Gerrit Code Review System

Change 82112 - Needs Code-Review

DESCRIPTION

SoftAVCEncoder: Support Surface input

This is pretty much based on a0a63e13788 by Andreas Huber.

The same restrictions as mentioned in that commit apply:

This relies heavily on the fact that the "Nexus" devices use ARGB32 as the color space for the data underlying a surface provided by SurfaceFlinger (mirroring). Generally there are no such guarantees.

Change-Id: I518fa2c3107e314d1912c60d1cdad742ab0a2e2

MESSAGES

+60, -5

Owner Martin Storsjö
Reviewers Deckard Autoverifier
Glenn Kasten Lajos Molnar
Project platform/frameworks/av
Branch master
Topic avcenc-surface-input
Strategy Always Merge
Updated 54 minutes ago

REVIEWERS

Martin Storsjö
Uploaded patch set 1.

Deckard Autoverifier
Patch Set 1: Verified-1 This is an automated message from the Deckard Autoverifier. Patchset merges and builds in A

Deckard Autoverifier
Patch Set 1: Verified

Martin Storsjö
Patch Set 1: Lajos, this is the patch for adding support for Surface input to the SW H264 encoder. This is quite similar

Deckard Autoverifier
Patch Set 1: Verified-1 This is an automated message from the Deckard Autoverifier. Patchset merges and builds in A
Measuring the scale and complexity of a Software Engineering (SE) activity is required for computing the contribution and performance of developers and also for effort and cost measurement models.

Patches submitted for code inspection to a peer code review system can vary widely in terms of size (number of files and code churn) and complexity (program complexity). Several variables such as developer expertise, time taken to accomplish a task and size and complexity of a task is required to develop effort estimation model.
Peer Code Review (PCR) Systems do not record or maintain the size and complexity of the reported issues.

**Research Aim**

- To investigate *metrics for measuring a code reviewer’s effort and contribution* based on the size and complexity of the modified files and the submitted patch.
- To conduct a *survey of practitioners* involved in peer code review activity to gather information and test our hypothesized causal relationship.

*Our aim is to conduct a survey of practitioners involved in peer code review activity to gather information and test our hypothesized causal relationship.*
## Survey Results of 44 Google Android Project Members Engaged in Peer Code Review Activity

<table>
<thead>
<tr>
<th>Q1: How many years of experience do you have with peer code review systems, code inspection, patch submission and review?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less than 1 year</td>
</tr>
<tr>
<td>Between 1 and 3 years</td>
</tr>
<tr>
<td>Between 3 and 5 years</td>
</tr>
<tr>
<td>More than 5 years</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q2: Patch review and code inspection requires program comprehension and understanding. Please evaluate the following size and complexity measures in terms of their suitability in computing the effort required to review a patch and contribution made by a patch reviewer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of issues reviewed</td>
</tr>
<tr>
<td>Code churn which is the sum of the number of lines added and deleted in a patch</td>
</tr>
<tr>
<td>Number of classes in the modified source code files</td>
</tr>
<tr>
<td>Number of functions in the modified source code files</td>
</tr>
<tr>
<td>Number of non-commenting source statements in the modified source code files</td>
</tr>
<tr>
<td>Average non-commenting source statement per function in the modified source code files</td>
</tr>
<tr>
<td>Average cyclomatic complexity per function in the modified source code files</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q3: More complex patches take more time to get accepted in general.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completely agree</td>
</tr>
<tr>
<td>Somewhat agree</td>
</tr>
<tr>
<td>Neither disagree nor agree</td>
</tr>
<tr>
<td>Somewhat disagree</td>
</tr>
<tr>
<td>Completely disagree</td>
</tr>
</tbody>
</table>
Important Take-Away from the Survey

43.18% of developers have more than 5 years of experience

Two most important factors in measuring effort

- Size of the code churn
- Average Cyclomatic Complexity per function in the modified source code files
Related Work

Kanij et al. mention that currently no standard methods for assessing the performance of software testers (or standard assessment criteria) are present [2012]

Kaner et al. presents a multidimensional, multi-sourced and qualitative approach to evaluate individual testers [2003]

Rastogi et al. present several metrics to measure contribution and performance of bug fixers, bug reporters and triagers by mining activity data from issue tracking system [2013]

Gousios et al. propose a model by combining traditional contribution metrics with data mined from software repositories for the purpose of accurately measuring developer contributions [2008]

Nagwani et al. present a team member ranking approach by mining attributes (such as severity and priority of bugs, number of bugs fixed and comments made by developers) available in software bug repositories [2012]

Ahsan et al. present an approach of mining effort data from the history of the developer’s bug fix activities and present an approach to build an effort estimation model for Open Source Software [2010]
Effort and Contribution Metrics

\[
EFF = \left( \frac{ACF}{ANF} \right)^\alpha (W_{CHN}.CHN + W_{NCS}.NCS) \quad (1)
\]

\[
EFF = \left( \frac{ACF}{ANF} \right)^\alpha (W_{CHN}.CHN + W_{NFL}.NFL) \quad (2)
\]

<table>
<thead>
<tr>
<th>CHN</th>
<th>Code churn which is the sum of the number of lines added and deleted in a patch</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCL</td>
<td>Number of classes in the modified source code files</td>
</tr>
<tr>
<td>NFL</td>
<td>Number of functions in the modified source code files</td>
</tr>
<tr>
<td>NCS</td>
<td>Number of non-commenting source statements in the modified source code files</td>
</tr>
<tr>
<td>ANF</td>
<td>Average non-commenting source statement per function in the modified source code files</td>
</tr>
<tr>
<td>ACF</td>
<td>Average cyclomatic complexity per function in the modified source code files</td>
</tr>
</tbody>
</table>
Empirical Study on Google Android Dataset

- Google Android is an open-source project which uses Gerrit peer code review system
- We download 1934 issues from issue id 20107 to 38729 (from January 2011 to July 2012) containing Java source files
- Table shows the descriptive statistics for six variables in the dataset consisting of 1934 issues

<table>
<thead>
<tr>
<th></th>
<th>ANF</th>
<th>ACF</th>
<th>NCS</th>
<th>NFL</th>
<th>NCL</th>
<th>CHN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Q1</td>
<td>7.085</td>
<td>2.18</td>
<td>185</td>
<td>16</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>Q2</td>
<td>9.065</td>
<td>3.355</td>
<td>510</td>
<td>49</td>
<td>6</td>
<td>31</td>
</tr>
<tr>
<td>Q3</td>
<td>12.185</td>
<td>4.4</td>
<td>1184</td>
<td>111</td>
<td>18</td>
<td>174</td>
</tr>
<tr>
<td>Max</td>
<td>156</td>
<td>51</td>
<td>26848</td>
<td>2240</td>
<td>349</td>
<td>44361</td>
</tr>
</tbody>
</table>

- Results reveals wide variability and dispersion in the dataset
- Two issues can have substantial difference between them in terms of size and complexity
Conclusions

- We propose six variables (and a metric) across three dimensions to measure the size and complexity of a patch review task.
- We conduct a survey of Google Android project developers to validate the suitability of proposed effort variables.
- We perform an empirical analysis on Google Android peer code review dataset and apply the proposed metrics to compute reviewer effort and contribution.