8-14-2015

Software Metrics and Dashboard

Shilpika Shilpika
sshilpika@luc.edu

George K. Thiruvathukal
Loyola University Chicago, gkt@cs.luc.edu

Saulo Aguiar
Loyola University Chicago, saguiar@luc.edu

Konstantin Läufer
Loyola University Chicago, klaeufer@gmail.com

Nicholas J. Hayward
Loyola University Chicago, nhayward@luc.edu

Recommended Citation
Shilpika; Thiruvathukal, George K.; Aguiar, Saulo; Läufer, Konstantin; and Hayward, Nicholas J., Software Metrics and Dashboard (2015). Retrieved from Loyola eCommons, Computer Science: Faculty Publications and Other Works,
Software Metrics Dashboard

Shilpika, Saulo Aguilar, Dr. George K. Thiruvathukal, Dr. Konstantin Läufer, Dr. Nicholas Hayward
sshilpika@luc.edu, saaguar@luc.edu, gkt@cs.luc.edu, laufer@cs.luc.edu, ancientlives@gmail.com

About Software Metrics

- Computed from one or more measured values
- A critical tool that provides continuous insight to products and processes
- Helps build reliable software in mission-critical environments
- The two types of metrics relevant to our work are:
  1. Computed from one or more measured values
  2. Developed to assess the effectiveness of the dashboard in terms of software quality and software process.

Approach

1. Evaluate whether CSE teams find the metrics dashboard useful.
2. Develop a metrics dashboard that will work for teams using tools like Github, Bitbucket etc.
3. Assess the effectiveness of the dashboard in terms of project success and developer attitude towards metrics and process.

Preliminary Results

The metrics dashboard is developed using modern web development methodologies like Spray in Scala which provides client-server side REST/HTTP support on top of Akka.

Defect density for a given project in a Github repository is calculated as:

\[
\text{Defect Density} = \frac{\sum \text{Issues}}{\sum \text{KLOC}}
\]

Future Work

We will ensure the metrics dashboard is properly instrumented to allow actual usage of the tools to be determined as projects collectively take advantage of them.

Conclusions

Building appropriate sets of metrics, presented in a useful way, can prove beneficial to CSE software teams, large, small and solo.

Facilitating the production of quality software would be a key component for developing and sustain CSE software, especially as other mission-critical projects grow to depend on it.

In this project, we take steps to introduce a pragmatic set of metrics into CSE software projects by conducting surveys, building a metrics dashboard, and performing analysis and post-surveys on selected projects.

For interactive exploration of Metrics information and reduction in computation overhead the datasets are partitioned into clusters in a distributed environment which introduces concurrency and independent failure/recovery of partitioned tasks.

Future Work

We will ensure the metrics dashboard is properly instrumented to allow actual usage of the tools to be determined as projects collectively take advantage of them.

We plan to identify a set of metrics that are helpful to our own project (and for dissemination beyond scientific software teams in the future).

We will work toward a plugin framework, so teams can extend the dashboard with additional metrics we have not implemented yet.

Year 2 will end with a formal release of the metrics dashboard and online user manual.

Acknowledgements:

This material is based upon work supported by the National Science Foundation under Grant No. 1465347.

Bibliography

- Spray and Akka Documentation

Work Plan

The work plan is to perform the following activities:

1. Assess how metrics are used and which general classes/types of metrics will be useful in CSE (Computational Science and Engineering) projects.
2. Develop a metrics dashboard that will work for teams using tools like Github, Bitbucket etc.
3. Assess the effectiveness of the dashboard in terms of project success and developer attitude towards metrics and process.

Code Base

- https://github.com/LoyolaChicagoMetrics/loyolachicagometrics
- https://github.com/sshilpika/metrics-dashboard-commit-density
- https://github.com/SauloAguilar/metrics-dashboard-commit-density

Defect Density = Number of Defects / Module Size

Defect density for a given project in a Github repository is calculated as:

\[
\text{Defect Density} = \frac{\sum \text{Issues}}{\sum \text{KLOC}}
\]

<table>
<thead>
<tr>
<th>Month</th>
<th>Defect Density</th>
<th>KLOC</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>January</td>
<td>0.035</td>
<td>0.007</td>
<td>0</td>
</tr>
<tr>
<td>March</td>
<td>0.067</td>
<td>0.005</td>
<td>0</td>
</tr>
<tr>
<td>May</td>
<td>0.125</td>
<td>0.007</td>
<td>0</td>
</tr>
<tr>
<td>Apr</td>
<td>0.095</td>
<td>0.006</td>
<td>0</td>
</tr>
<tr>
<td>Sep</td>
<td>0.067</td>
<td>0.006</td>
<td>0</td>
</tr>
<tr>
<td>Dec</td>
<td>0.085</td>
<td>0.006</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 1. Total commits per month (django/django)

Figure 2. Total cumulative commits per month (django/django)

Figure 3. Total commits per month (django/django)

Figure 4. Defect Density per month (shilpika/metrics-test)

Figure 5. Total commits per month (django/django)

Figure 6. Total cumulative commits per month (django/django)

Figure 7. Total commits per month (django/django)