Code Churn

Yegor Bugayenko

Lecture #13 out of 24
80 minutes

The slidedeck was presented by the author in this YouTube Video

History of Version Control Systems (VCS)

- The Librarian by ADR: created in 1969
- Panvalet by Computer Associates: 1969
- SCCS by Bell Labs: 1973
- RCS by GNU: 1982
- CVS: 1986
- ClearCase by IBM: 1992
- Apache Subversion: 2000
- Mercurial: 2005
- Git: 2005

Sebastian G. Elbaum

"We can measure the increase or decrease in system complexity as measured by a selected metric, code delta, or we can measure the total amount of change the system has undergone between builds, code churn."

— J.C. Munson and S.G. Elbaum. Code Churn: A Measure for Estimating the Impact of Code Change. In *Proceedings of the Proceedings. International Conference on Software Maintenance (Cat. No. 98CB36272)*, 1998. doi:10.1109/icsm.1998.738486

Motivating Example

Commit #1:

0

```
class Book
  private final int id;
  public Book(int it)
    this.id = i;
```

Commit #2:

+5

```
class Book
  private int id;
  public Book(int it)
    this.id = i;
  public int getId()
    return this.id;
  private int setId(int i)
    this.id = i;
```

Commit #3:

+2 / -2

```
final class Book
  private final int id;
  public Book(int it)
    this.id = i;
  public int getId()
    return this.id;
```

Delta: +3 / 0, Churn: +7 / -2

Code Delta vs. Code Churn

BUILD	RCM	DELTA	CHURN
1	184643.38	5.72	7.12
2	184648.81	5.43	16.78
3	185702.10	1053.28	1547.93
4	185857.61	155.50	187.11
5	186236.58	378.97	429.30
6	186261.81	25.23	294.18
7	186083.21	-178.60	791.87
8	186726.70	643.50	836.71
9	187072.37	345.65	426.24
10	187157.08	84.71	86.74
11	189945.91	2788.85	2804.02
12	190000.81	54.89	59.28
13	190007.10	6.29	6.38
14	190012.49	5.39	5.48
15	190069.36	56.87	75.08
16	190066.48	-2.88	3.86
17	190067.65	1.17	1.17
18	190067.81	0.16	0.21

Table 2: Evolution Data for QTB

"A limitation of measuring code deltas is that it doesn't give an indicator as to how much change the system has undergone. If several software modules are removed and are replaced by modules of roughly equivalent complexity, the code delta for the system will be close to zero."

Source: *Code Churn: A Measure for Estimating the Impact of Code Change*, Sebastian G. Elbaum et al., ICSM, 1998

Nachiappan Nagappan

"Our case study provides strong support for the following conclusion: increase in relative code churn measures is accompanied by an increase in system defect density."

— Nachiappan Nagappan and Thomas Ball. Use of Relative Code Churn Measures to Predict System Defect Density. In *Proceedings of the 27th International Conference on Software Engineering*, pages 284–292, 2005. doi:10.1145/1062455.1062514

Rudolf Ferenc

"We can conclude that committing files with higher cumulative code churn values (i.e. those of longer change history) is more likely to result in negative maintainability change."

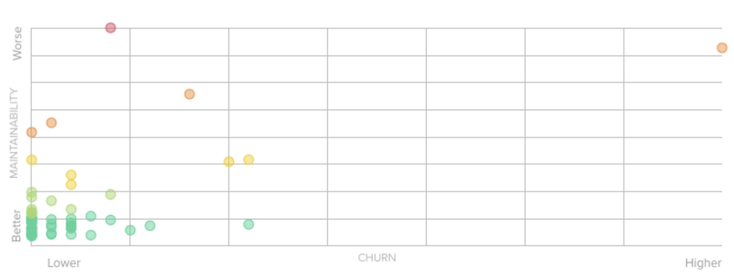
— Csaba Faragó, Péter Hegedűs, and Rudolf Ferenc. Cumulative Code Churn: Impact on Maintainability. In *Proceedings of the 15th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, pages 141–150. IEEE, 2015. doi:10.1109/scam.2015.7335410

Tobias Olsson

"Non-normal code churn can be a tangible and measurable effect of architectural violations in source code. However, more research is needed to better understand this connection, e.g., if violations are the cause, the size of the effect, the cost of refactoring, and whether it is generalizable to other system."

— Tobias Olsson, Morgan Ericsson, and Anna Wingkvist. The Relationship of Code Churn and Architectural Violations in the Open Source Software JabRef. In *Proceedings of the 11th European Conference on Software Architecture: Companion Proceedings*, 2017. doi:10.1145/3129790.3129810

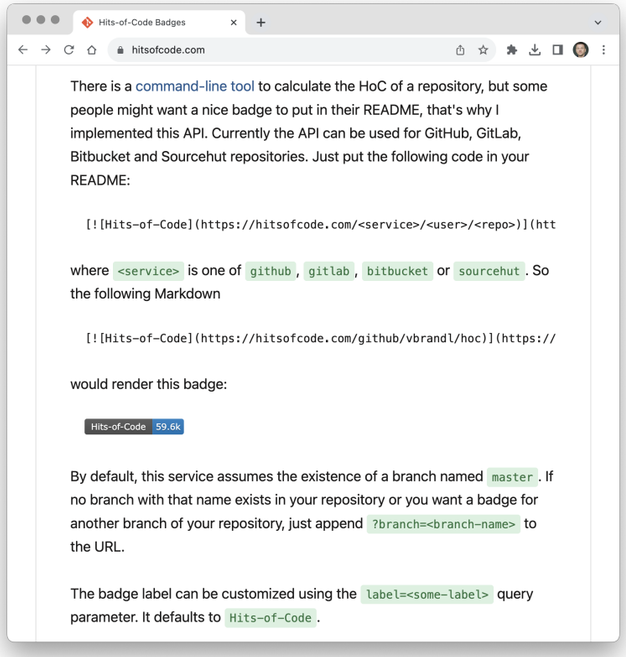
CodeClimate.com Interpretation of Code Churn



"When we have finished analyzing your default branch, you will see churn metrics for the files in your repository. This churn metric is approximately the number of times a file has changed in the last 90 days, calculated by counting the number of distinct versions of that file across commits from that time."

Source: their blog

HitsOfCode.com



Created by Valentin Brandl (@vbrandl) and inspired by the blog post of mine: Hits-of-Code Instead of SLoC (2014)

Read this:

Hits-of-Code Instead of SLoC (2014)

References

Csaba Faragó, Péter Hegedűs, and Rudolf Ferenc. Cumulative Code Churn: Impact on Maintainability. In *Proceedings of the 15th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, pages 141–150. IEEE, 2015. doi:10.1109/scam.2015.7335410.

J.C. Munson and S.G. Elbaum. Code Churn: A Measure for Estimating the Impact of Code Change. In *Proceedings of the Proceedings. International Conference on Software Maintenance (Cat. No. 98CB36272)*, 1998. doi:10.1109/icsm.1998.738486.

Nachiappan Nagappan and Thomas Ball. Use of Relative Code Churn Measures to Predict System Defect Density. In *Proceedings of the 27th International Conference on Software Engineering*, pages 284–292, 2005. doi:10.1145/1062455.1062514.

Tobias Olsson, Morgan Ericsson, and Anna Wingkvist. The Relationship of Code Churn and Architectural Violations in the Open Source Software JabRef. In *Proceedings of the 11th European Conference on Software Architecture: Companion Proceedings*, 2017. doi:10.1145/3129790.3129810.