OSS2018 – 14th International Conference on Open Source Systems
June 8-10, 2018,
Harokopio University, Athens, Greece

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

Commit Quality Measures	Commit Score					
	1	2	3	4	5	unit
Length of Title	=0 or >72	1-10	11-30	31-50	51-72	number of characters
Title ends with dots	No	Yes				y→1, n→0
Title first character capital	No	Yes				y→1, n→0
Count number of "and" "or" in Title	>6	5-6	3-4	1-2	0	count
Count number of "file name" in title	>6	5-6	3-4	1-2	0	count
Count number of external references in title	>6	5-6	3-4	1-2	0	count
Imperative mode in title	No	Yes				y→1, n→0
Commit body existence	No	Yes				y→1, n→0
Count number of "file name" in body	0	10>	6-10	3-5	1-2	count
Count number of external references in body	0	10>	6-10	3-5	1-2	count
Count number of paragraph in body	0	10>	5-10	3-4	1-2	count

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

OSS Projects	Origin Date	Number of Months	Number of Contributors	Commit messages	
PostgreSQL	Jul, 1996	239	43	54355	
glibc	Feb, 1989	321	410	43313	
Eclipse-CDT	Jun, 2002	168	203	28817	
GnuCash	Nov, 1997	222	105	21969	
WordPress	Apr, 2003	158	73	37333	
Firebug	Aug, 2007	181	45	13043	
Rhino	Apr, 1999	105	56	3721	

Analyzing the quality of commit messages in the OSS projects

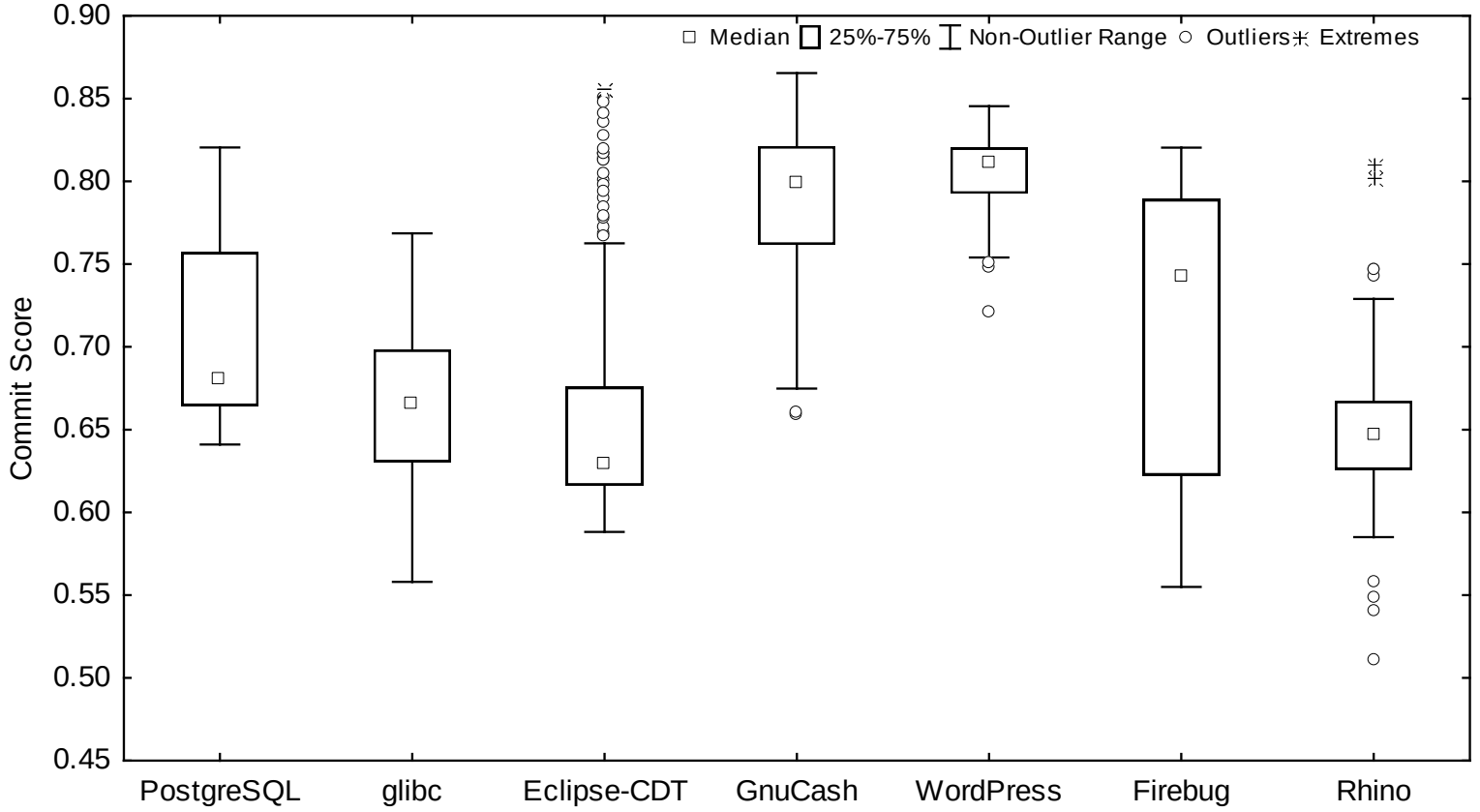
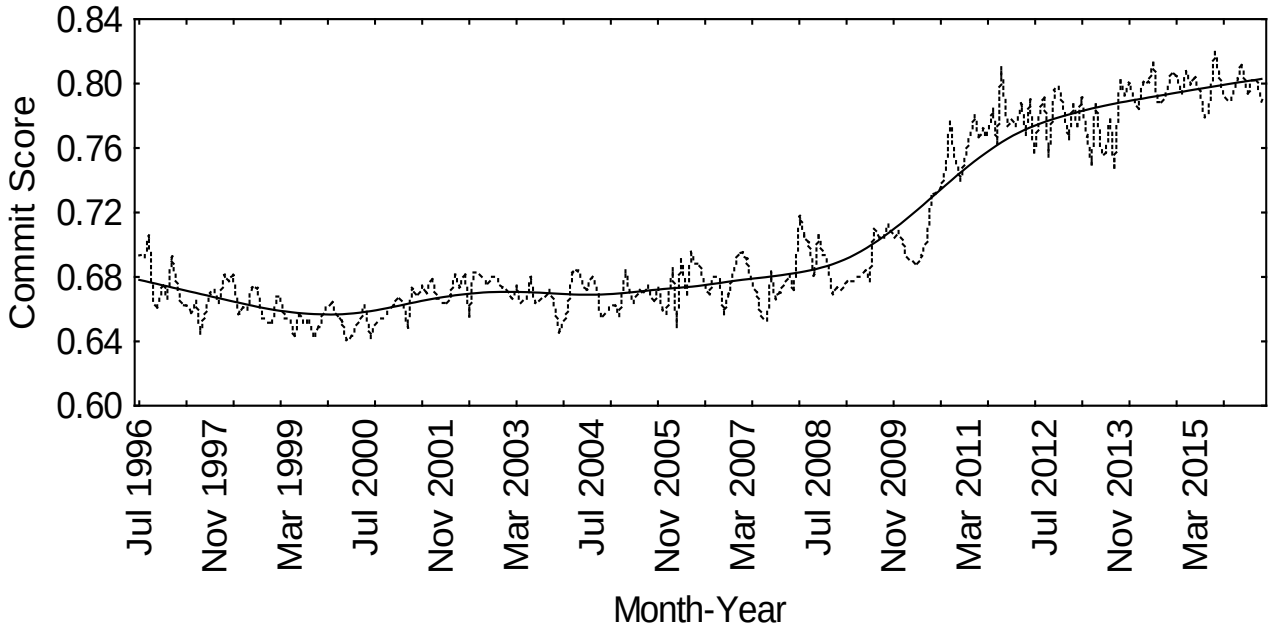
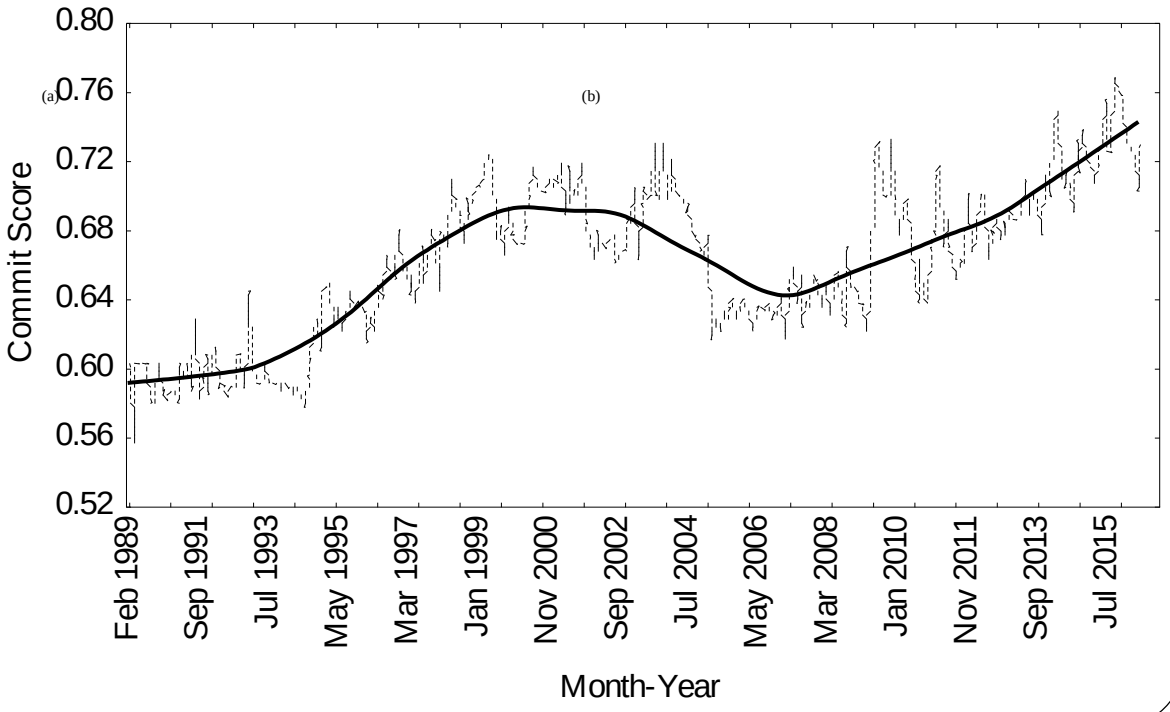


Fig. 1. Variation in commit quality of the OSS projects



PostgreSQL

glibc



Commit Score
0.88
0.84
0.80
0.76
0.72
0.68
0.64
0.60
0.56
Jun 2002
Jun 2003
Jun 2004
Jun 2005
Jun 2006
Jun 2007
Jun 2008
Jun 2009
Jun 2010
Jun 2011
Jun 2012
Jun 2013
Jun 2014
Jun 2015
Month-Year

Commit Score
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
0.72
0.70
0.68
0.66
0.64
Nov 1997
Feb 1999
Jun 2000
Sep 2001
Dec 2002
Mar 2004
Jun 2005
Sep 2006
Dec 2007
Mar 2009
Jun 2010
Sep 2011
Dec 2012
Mar 2014
Jun 2015
Month-Year

0.88
0.84
0.80
0.76
0.72
0.68
Commit Score
Apr 2003
Mar 2004
Feb 2005
Jan 2006
Dec 2006
Nov 2007
Oct 2008
Sep 2009
Aug 2010
Jul 2011
Jun 2012
May 2013
Apr 2014
Mar 2015
Feb 2016
Month-Year

0.84
0.80
0.76
0.72
0.68
0.64
0.60
0.56
0.52
Commit Score
Aug 2007
Apr 2008
Dec 2008
Aug 2009
Apr 2010
Dec 2010
Aug 2011
Apr 2012
Dec 2012
Aug 2013
Apr 2014
Dec 2014
Aug 2015
May 2016
Month-Year

Does the number of contributors affect the commit message syntactic quality?

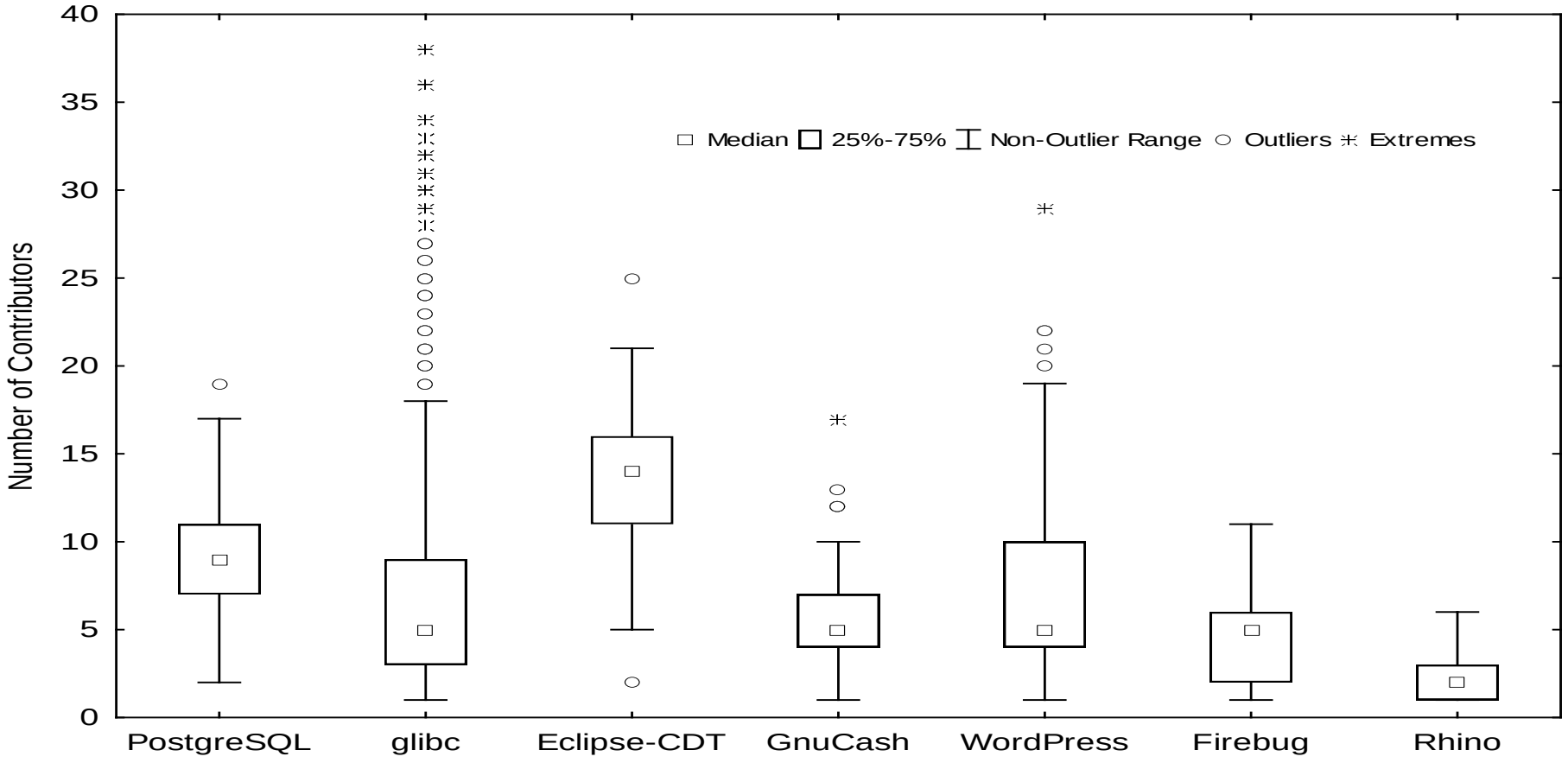
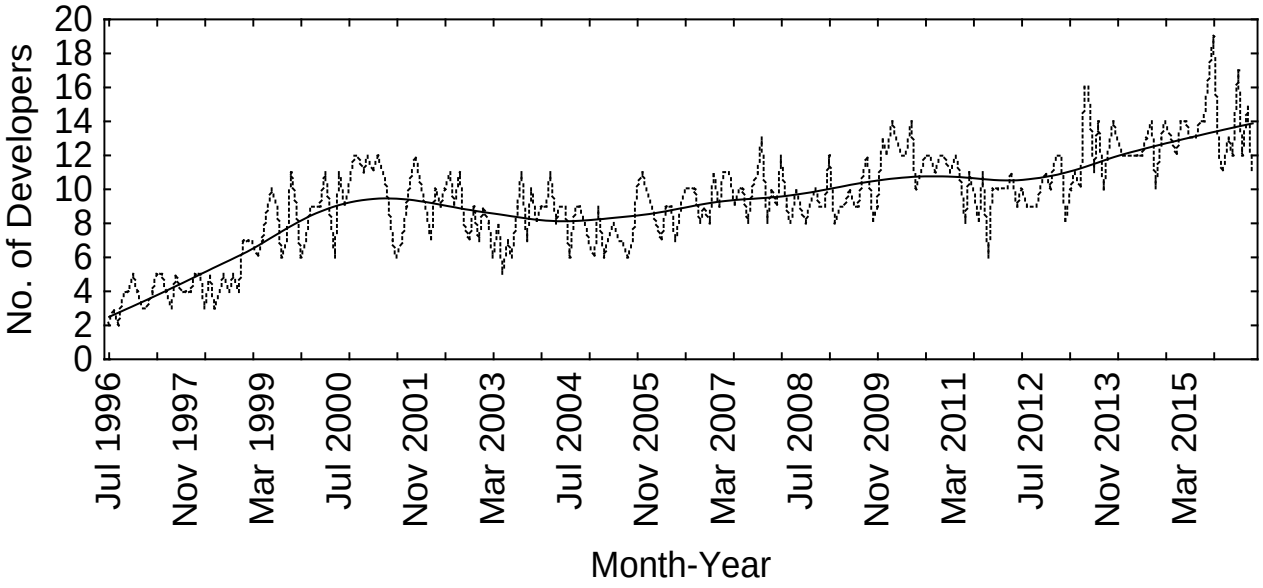


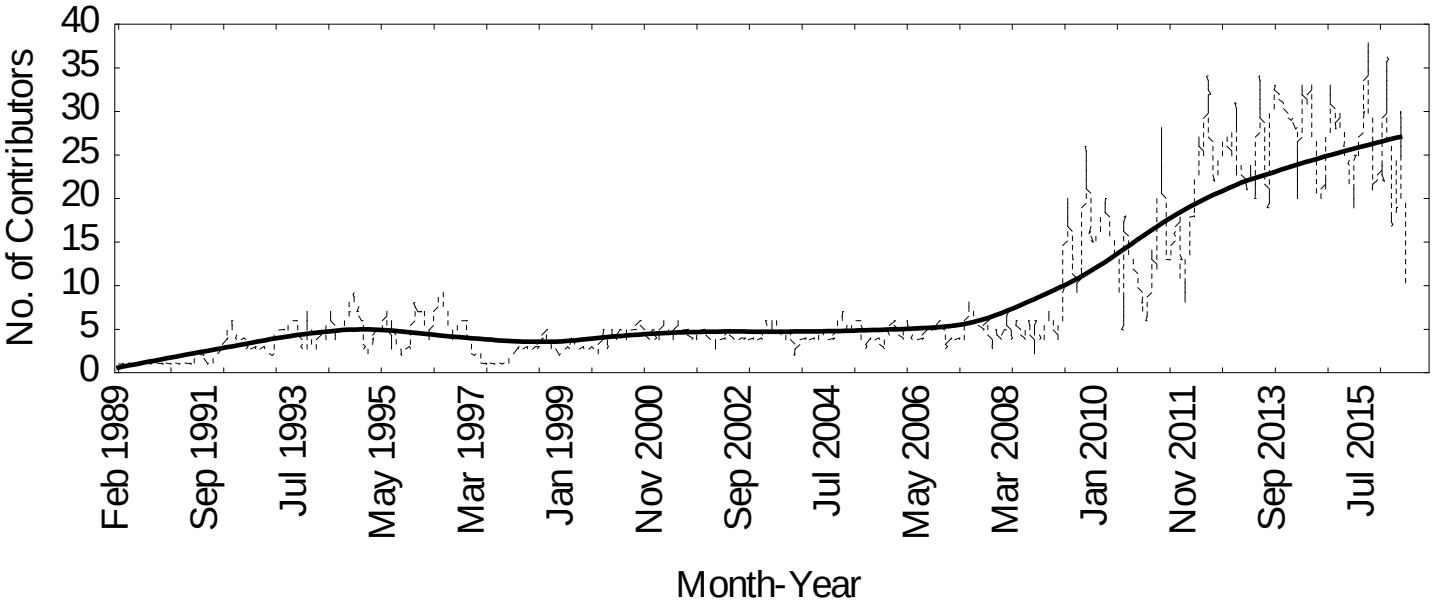
Fig. 3. Variation in the number of contributors of the OSS projects

Contributor churn of the OSS projects over the period of time



PostgreSQL

glibc



No. of Contributors
0
4
8
12
16
20
24
Jun 2002
Jun 2003
Jun 2004
Jun 2005
Jun 2006
Jun 2007
Jun 2008
Jun 2009
Jun 2010
Jun 2011
Jun 2012
Jun 2013
Jun 2014
Jun 2015
Month-Year

No. of Contributors
0
2
4
6
8
10
12
14
16
18
Nov 1997
Feb 1999
Jun 2000
Sep 2001
Dec 2002
Mar 2004
Jun 2005
Sep 2006
Dec 2007
Mar 2009
Jun 2010
Sep 2011
Dec 2012
Mar 2014
Jun 2015
Month-Year

No. of Contributors
0
5
10
15
20
25
30
35
Apr 2003
Mar 2004
Feb 2005
Jan 2006
Dec 2006
Nov 2007
Oct 2008
Sep 2009
Aug 2010
Jul 2011
Jun 2012
May 2013
Apr 2014
Mar 2015
Feb 2016
Month-Year

No. of Contributors
0
2
4
6
8
10
12
Aug 2007
Apr 2008
Dec 2008
Aug 2009
Apr 2010
Dec 2010
Aug 2011
Apr 2012
Dec 2012
Aug 2013
Apr 2014
Dec 2014
Aug 2015
May 2016
Month-Year

Understanding the Contribution Pattern

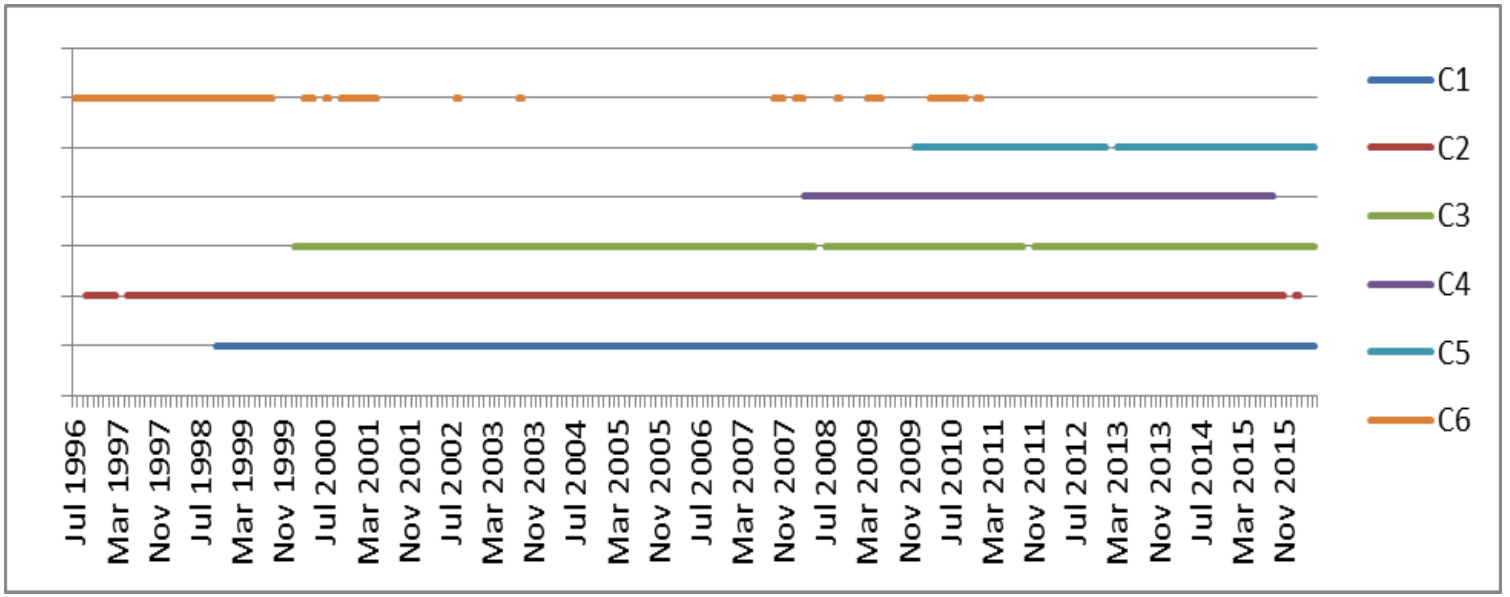
In this regard, the first step is to find commit distribution among different contributors of the OSS projects to identify the core group of contributors.

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 7. Contributor wise commits distribution (in %)

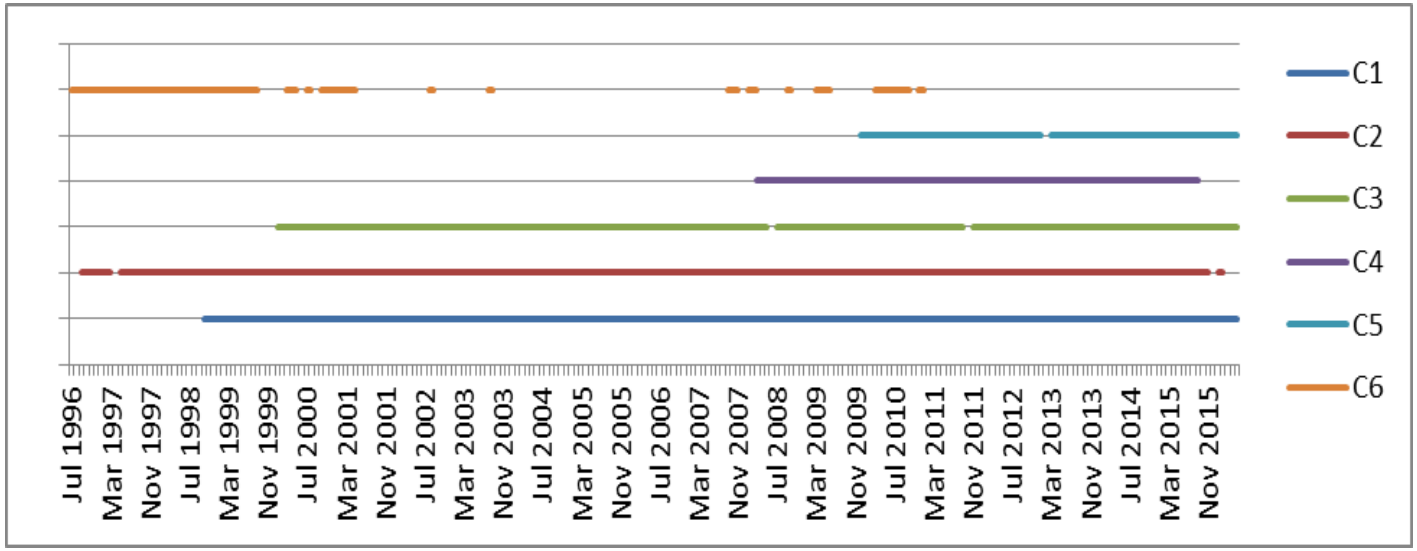
OSS Projects	C1	C2	C3	C4	C5	C6	Other
PostgreSQL	34.53	26.52	7.35	3.62	3.33	3.10	21.55
glibc	40.58	24.18	4.72	4.14	3.56	3.36	19.46
Eclipse-CDT	10.11	7.90	6.43	5.64	5.56	5.52	58.84
GnuCash	16.66	14.84	12.65	7.93	7.57	7.57	32.78
	22.1				4.7	3.8	

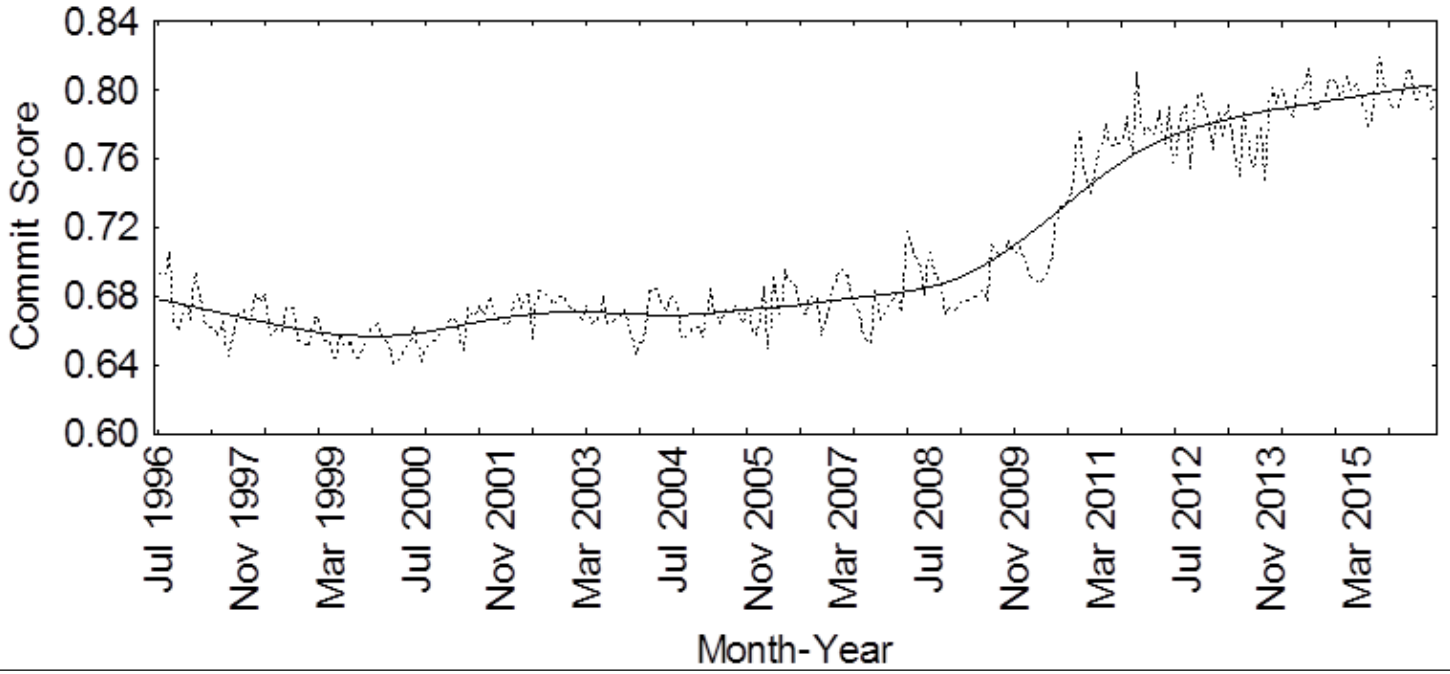
Commit regularity



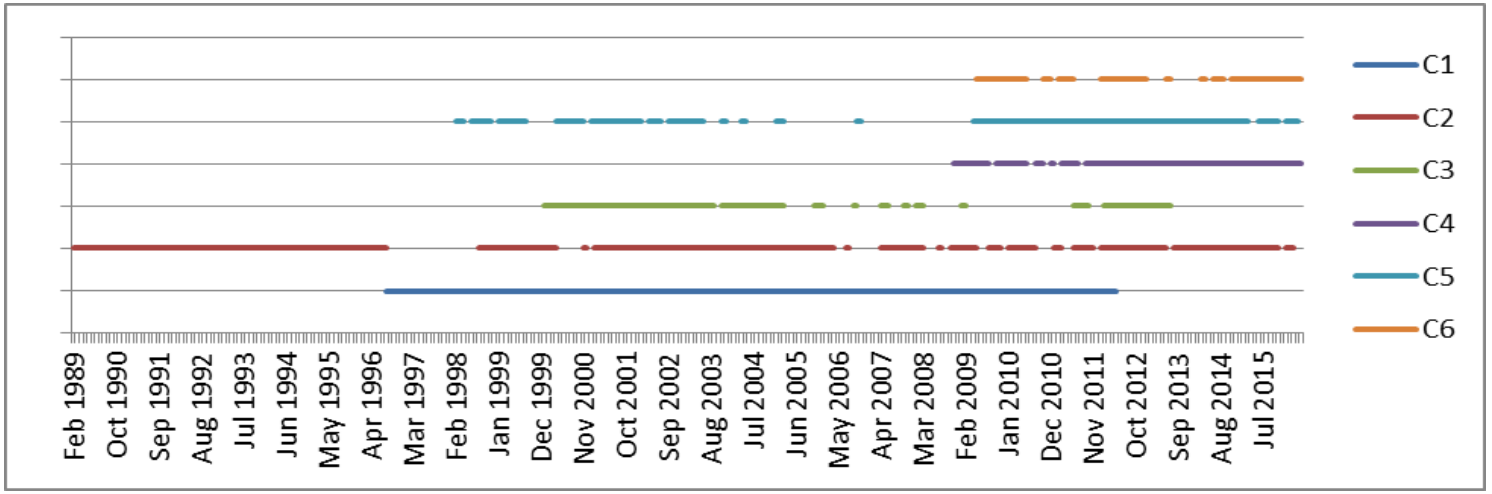
Analyzing Commit Regularity and Commit Quality

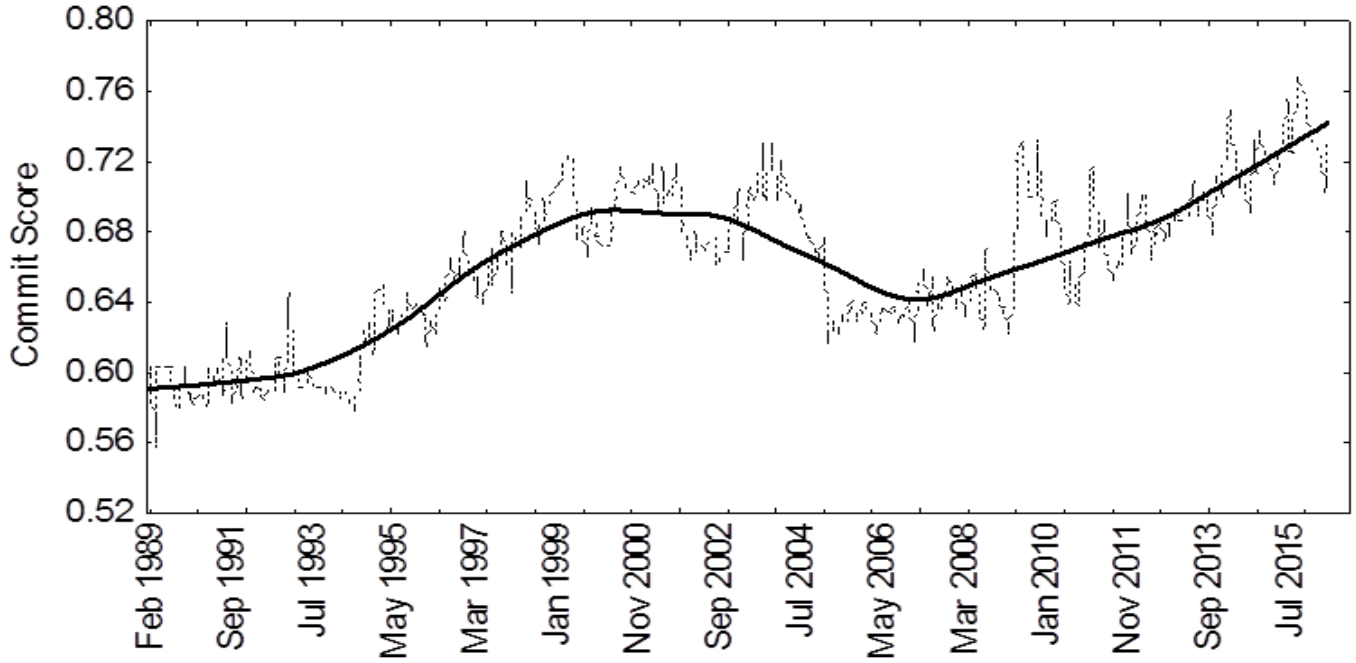
PostgreSQL


C1
C2
C3
C4
C5
C6
Jul 1996
Mar 1997
Nov 1997
Jul 1998
Mar 1999
Nov 1999
Jul 2000
Mar 2001
Nov 2001
Jul 2002
Mar 2003
Nov 2003
Jul 2004
Mar 2005
Nov 2005
Jul 2006
Mar 2007
Nov 2007
Jul 2008
Mar 2009
Nov 2009
Jul 2010
Mar 2011
Nov 2011
Jul 2012
Mar 2013
Nov 2013
Jul 2014
Mar 2015
Nov 2015

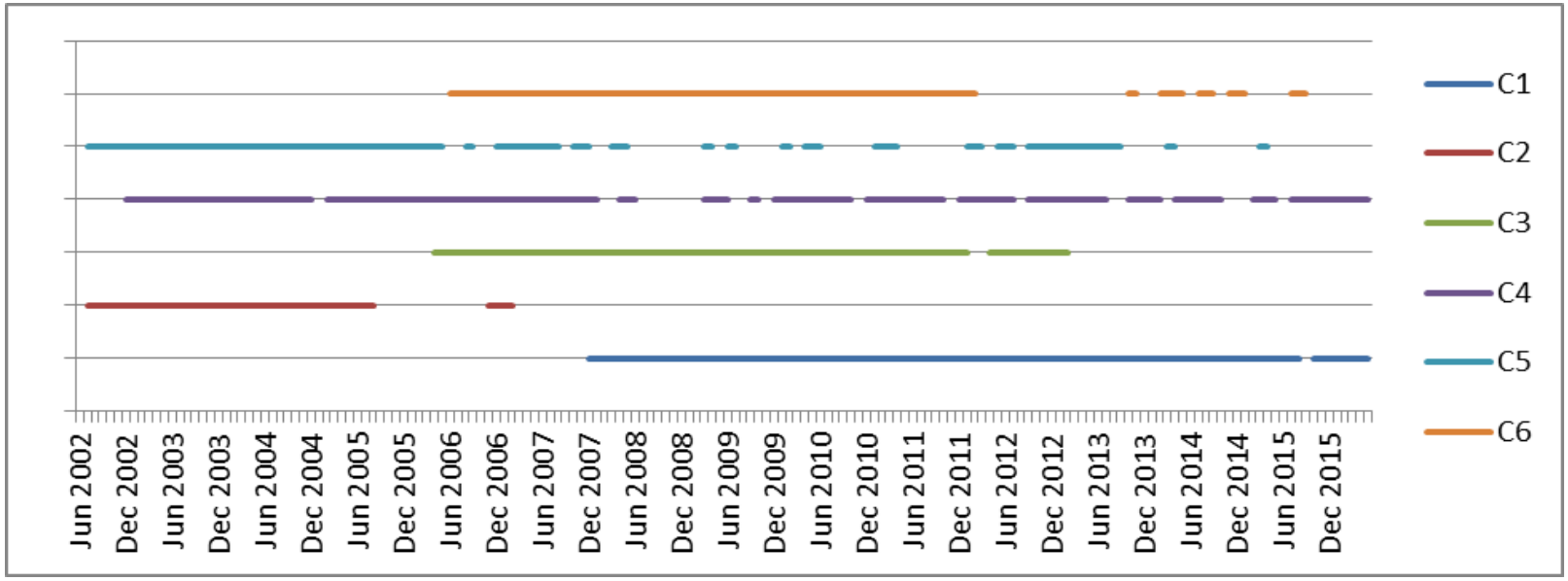

Commit Score
0.84
0.80
0.76
0.72
0.68
0.64
0.60
Jul 1996
Nov 1997
Mar 1999
Jul 2000
Nov 2001
Mar 2003
Jul 2004
Nov 2005
Mar 2007
Jul 2008
Nov 2009
Mar 2011
Jul 2012
Nov 2013
Mar 2015
Month-Year

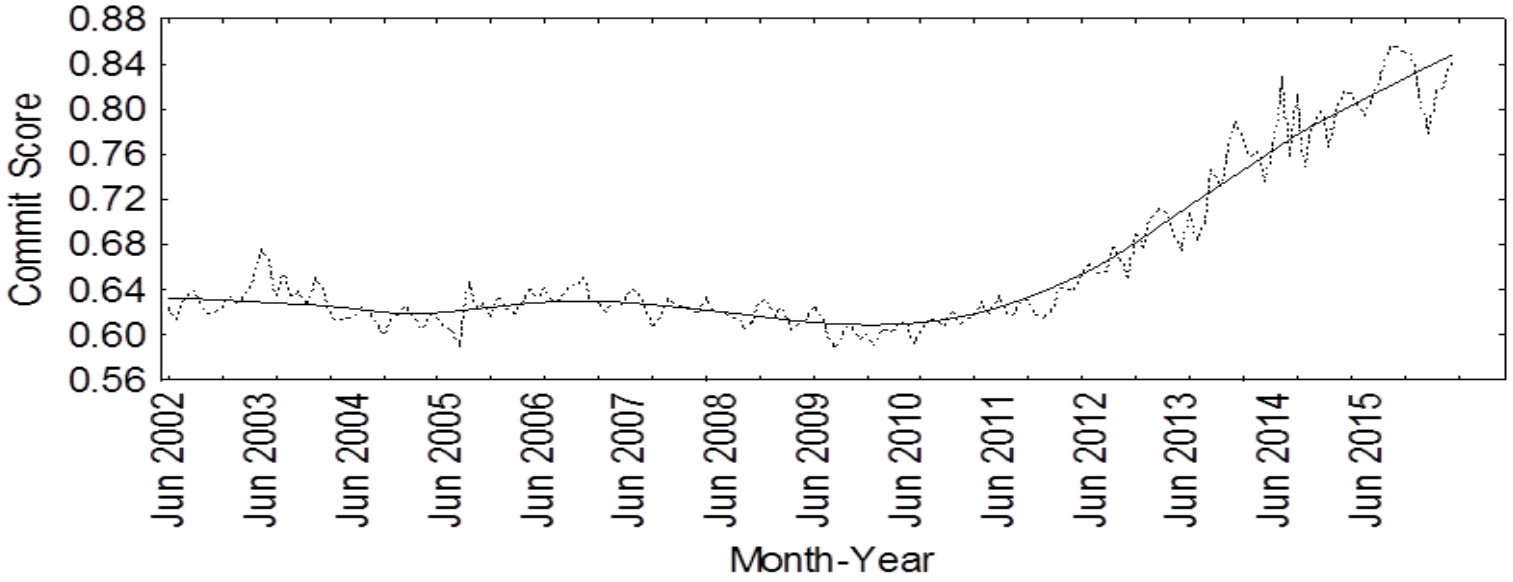
glibc


C1
C2
C3
C4
C5
C6
Feb 1989
Oct 1990
Sep 1991
Aug 1992
Jul 1993
Jun 1994
May 1995
Apr 1996
Mar 1997
Feb 1998
Jan 1999
Dec 1999
Nov 2000
Oct 2001
Sep 2002
Aug 2003
Jul 2004
Jun 2005
May 2006
Apr 2007
Mar 2008
Feb 2009
Jan 2010
Dec 2010
Nov 2011
Oct 2012
Sep 2013
Aug 2014
Jul 2015

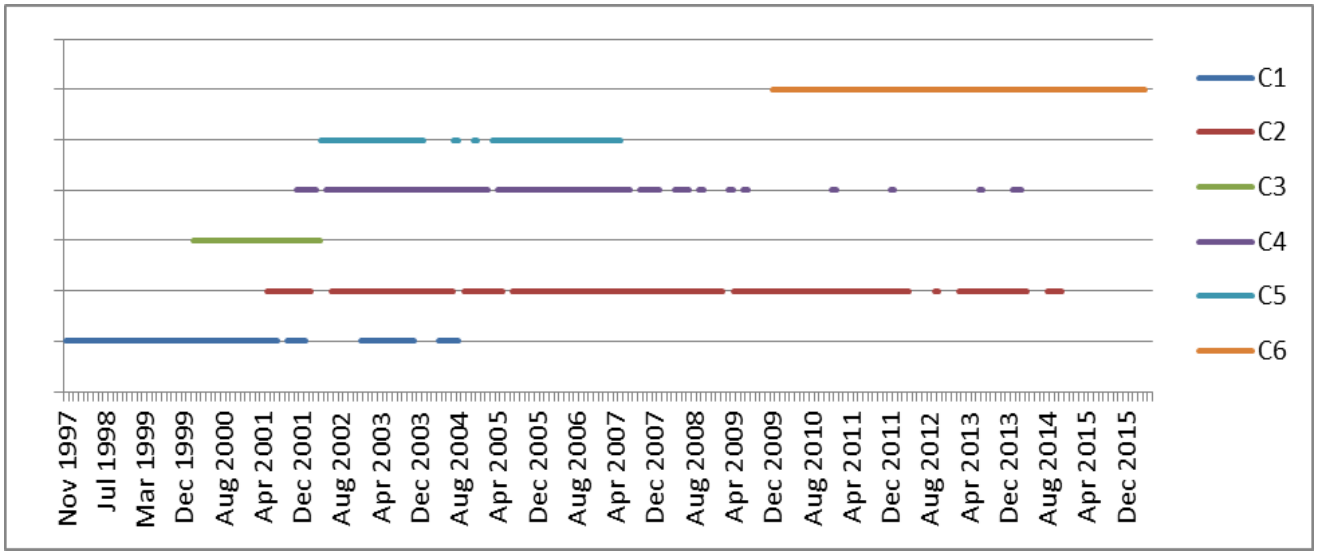

Commit Score
0.80
0.76
0.72
0.68
0.64
0.60
0.56
0.52
Feb 1989
Sep 1991
Jul 1993
May 1995
Mar 1997
Jan 1999
Nov 2000
Sep 2002
Jul 2004
May 2006
Mar 2008
Jan 2010
Nov 2011
Sep 2013
Jul 2015

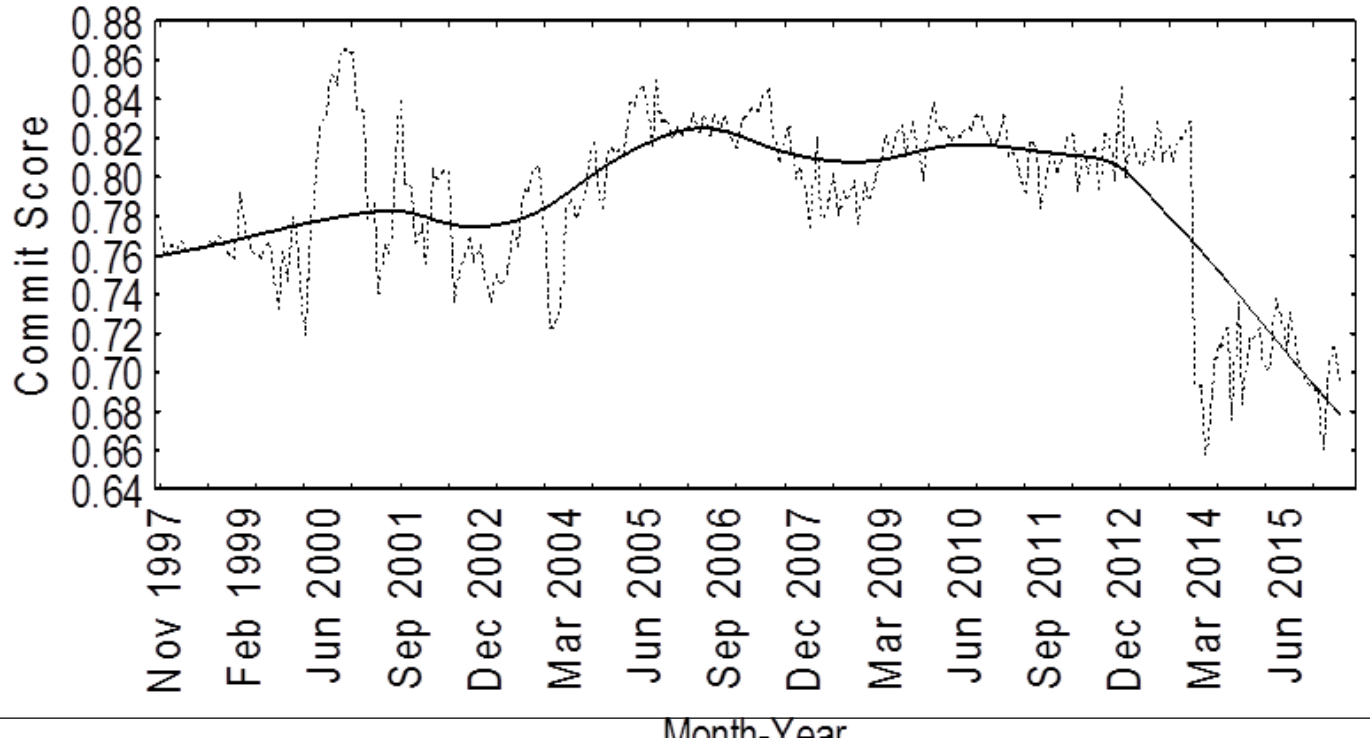
Eclipse-CDT


C1
C2
C3
C4
C5
C6
Jun 2002
Dec 2002
Jun 2003
Dec 2003
Jun 2004
Dec 2004
Jun 2005
Dec 2005
Jun 2006
Dec 2006
Jun 2007
Dec 2007
Jun 2008
Dec 2008
Jun 2009
Dec 2009
Jun 2010
Dec 2010
Jun 2011
Dec 2011
Jun 2012
Dec 2012
Jun 2013
Dec 2013
Jun 2014
Dec 2014
Jun 2015
Dec 2015

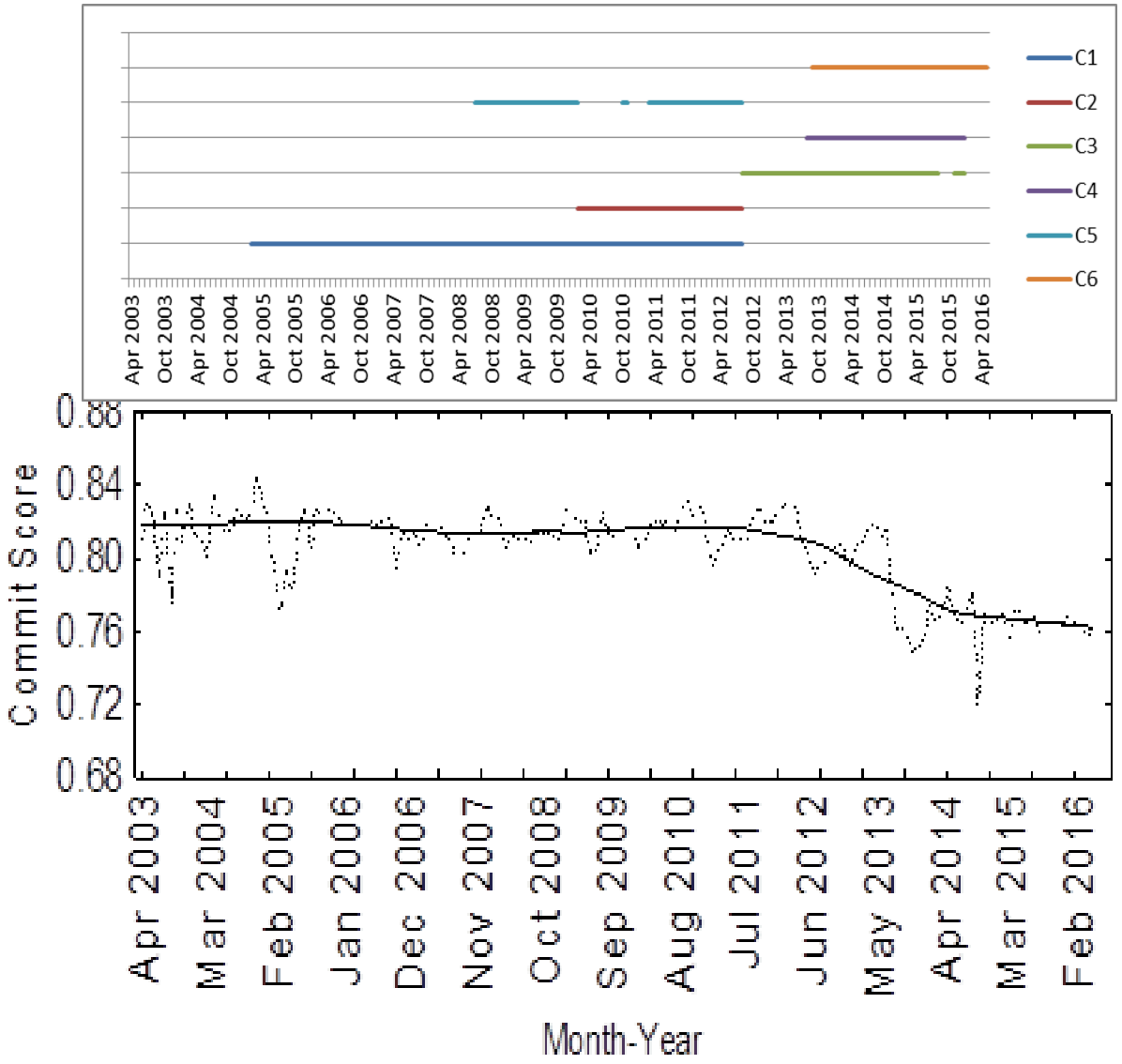

Commit Score
0.88
0.84
0.80
0.76
0.72
0.68
0.64
0.60
0.56
Jun 2002
Jun 2003
Jun 2004
Jun 2005
Jun 2006
Jun 2007
Jun 2008
Jun 2009
Jun 2010
Jun 2011
Jun 2012
Jun 2013
Jun 2014
Jun 2015
Month-Year

GnuCash

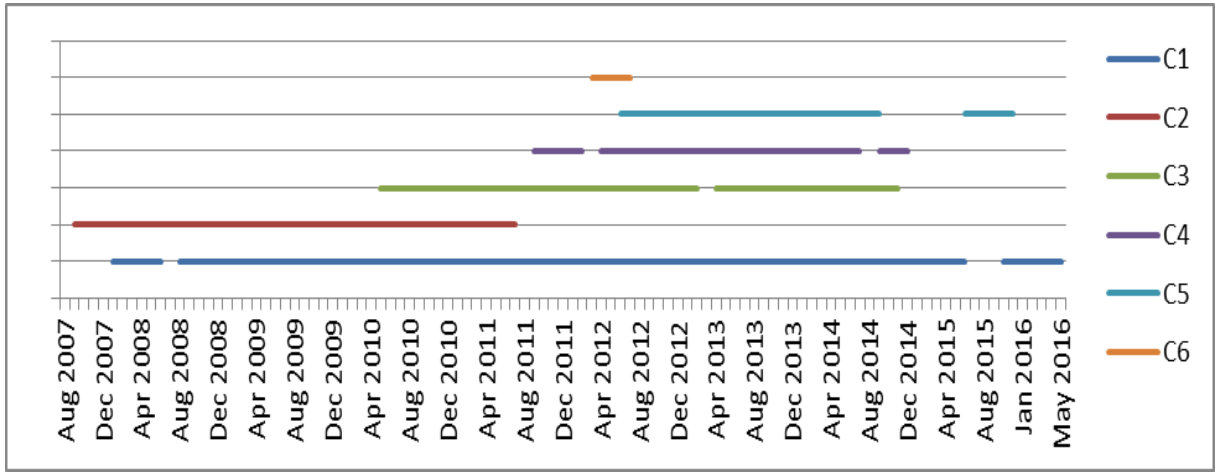

C1
C2
C3
C4
C5
C6
Nov 1997
Jul 1998
Mar 1999
Dec 1999
Aug 2000
Apr 2001
Dec 2001
Aug 2002
Apr 2003
Dec 2003
Aug 2004
Apr 2005
Dec 2005
Aug 2006
Apr 2007
Dec 2007
Aug 2008
Apr 2009
Dec 2009
Aug 2010
Apr 2011
Dec 2011
Aug 2012
Apr 2013
Dec 2013
Aug 2014
Apr 2015
Dec 2015

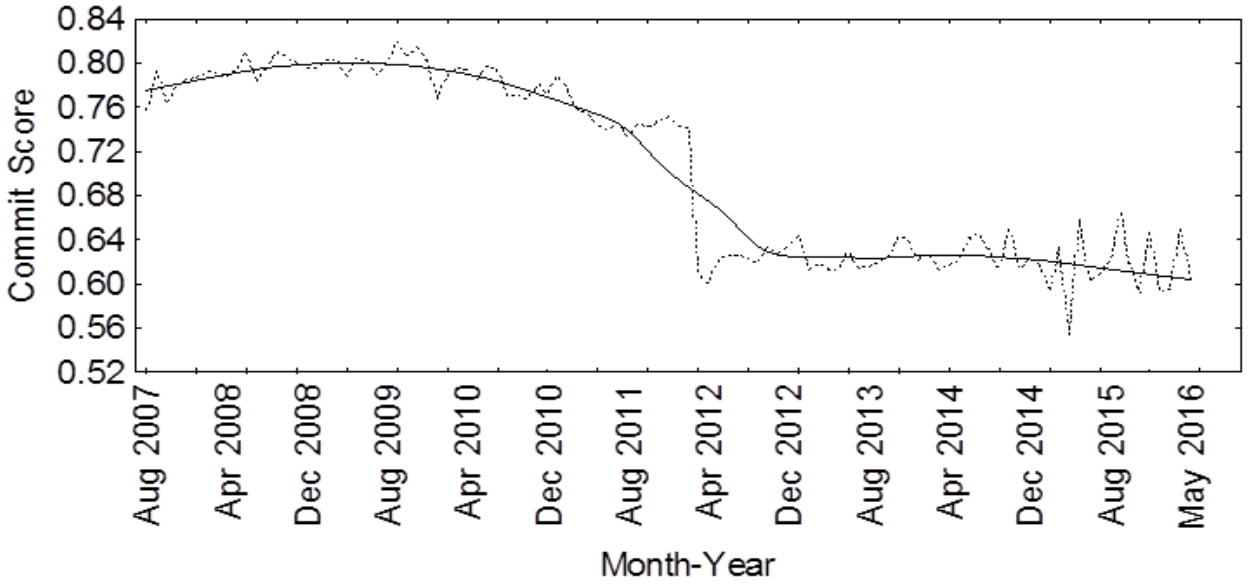

Commit Score
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0.74
0.72
0.70
0.68
0.66
0.64
Nov 1997
Feb 1999
Jun 2000
Sep 2001
Dec 2002
Mar 2004
Jun 2005
Sep 2006
Dec 2007
Mar 2009
Jun 2010
Sep 2011
Dec 2012
Mar 2014
Jun 2015
Month-Year

WordPress

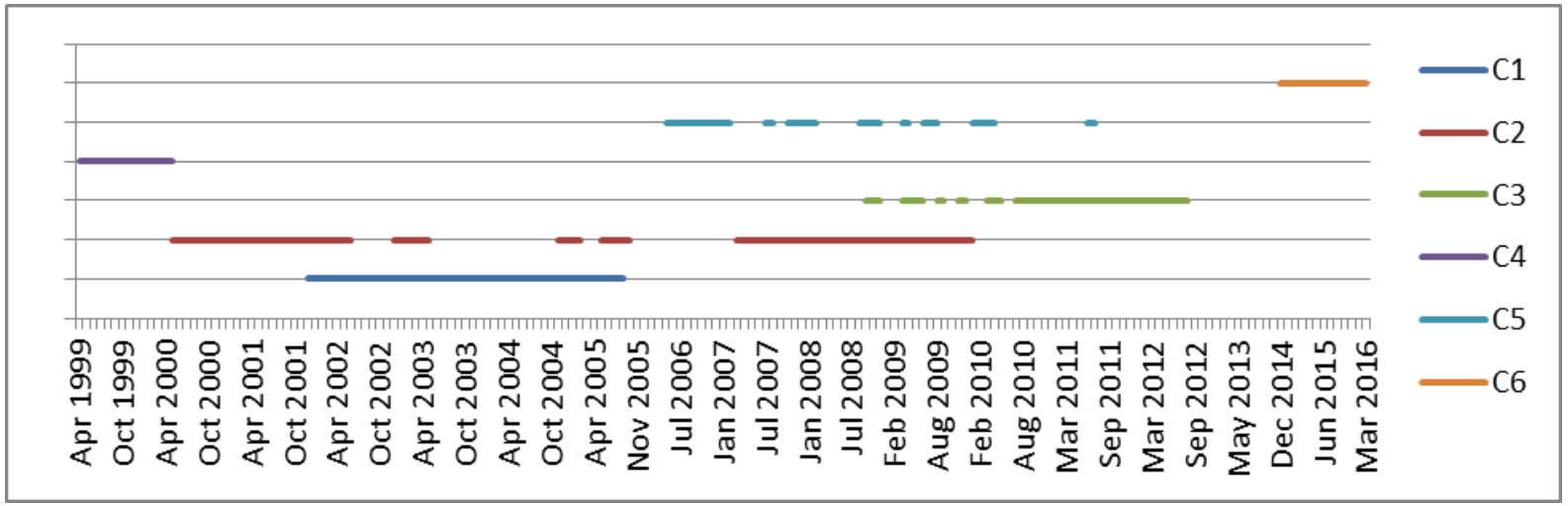

C1
C2
C3
C4
C5
C6
Apr 2003
Oct 2003
Apr 2004
Oct 2004
Apr 2005
Oct 2005
Apr 2006
Oct 2006
Apr 2007
Oct 2007
Apr 2008
Oct 2008
Apr 2009
Oct 2009
Apr 2010
Oct 2010
Apr 2011
Oct 2011
Apr 2012
Oct 2012
Apr 2013
Oct 2013
Apr 2014
Oct 2014
Apr 2015
Oct 2015
Apr 2016
Commit Score
0.88
0.84
0.80
0.76
0.72
0.68
Apr 2003
Mar 2004
Feb 2005
Jan 2006
Dec 2006
Nov 2007
Oct 2008
Sep 2009
Aug 2010
Jul 2011
Jun 2012
May 2013
Apr 2014
Mar 2015
Feb 2016
Month-Year

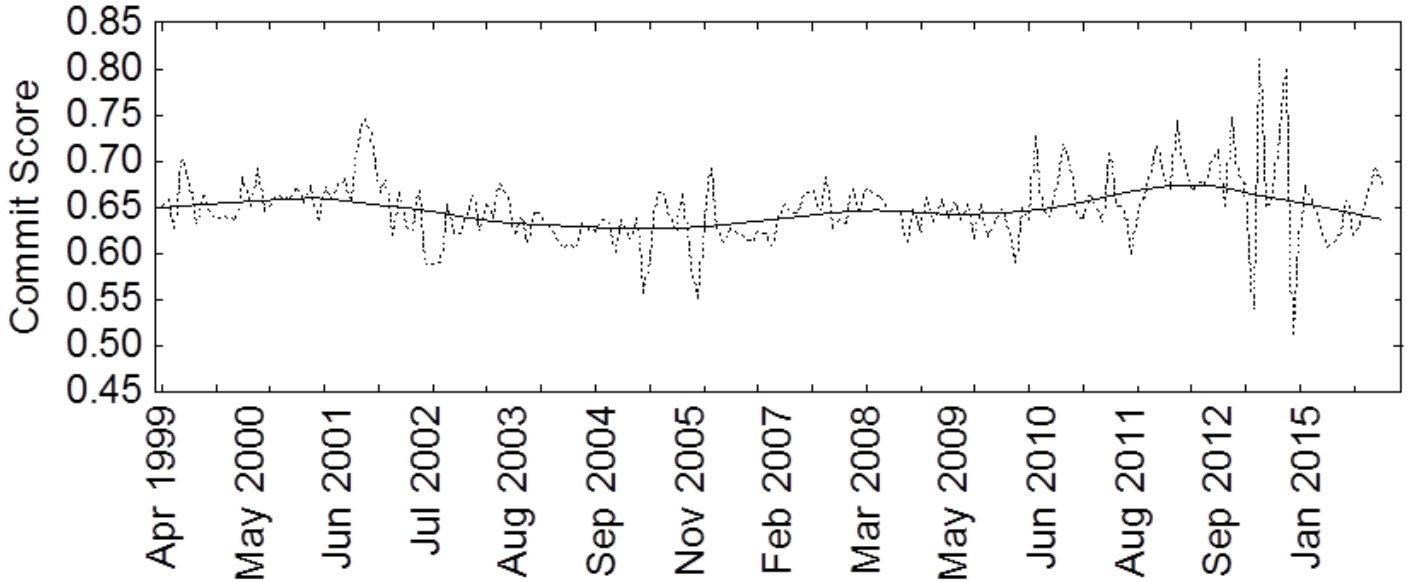
Firebug


C1
C2
C3
C4
C5
C6
Aug 2007
Dec 2007
Apr 2008
Aug 2008
Dec 2008
Apr 2009
Aug 2009
Dec 2009
Apr 2010
Aug 2010
Dec 2010
Apr 2011
Aug 2011
Dec 2011
Apr 2012
Aug 2012
Dec 2012
Apr 2013
Aug 2013
Dec 2013
Apr 2014
Aug 2014
Dec 2014
Apr 2015
Aug 2015
Jan 2016
May 2016


Commit Score
0.84
0.80
0.76
0.72
0.68
0.64
0.60
0.56
0.52
Aug 2007
Apr 2008
Dec 2008
Aug 2009
Apr 2010
Dec 2010
Aug 2011
Apr 2012
Dec 2012
Aug 2013
Apr 2014
Dec 2014
Aug 2015
May 2016
Month-Year

Rhino


C1
C2
C3
C4
C5
C6
Apr 1999
Oct 1999
Apr 2000
Oct 2000
Apr 2001
Oct 2001
Apr 2002
Oct 2002
Apr 2003
Oct 2003
Apr 2004
Oct 2004
Apr 2005
Nov 2005
Jul 2006
Jan 2007
Jul 2007
Jan 2008
Jul 2008
Feb 2009
Aug 2009
Feb 2010
Aug 2010
Mar 2011
Sep 2011
Mar 2012
Sep 2012
May 2013
Dec 2014
Jun 2015
Mar 2016


Commit Score
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
Apr 1999
May 2000
Jun 2001
Jul 2002
Aug 2003
Sep 2004
Nov 2005
Feb 2007
Mar 2008
May 2009
Jun 2010
Aug 2011
Sep 2012
Jan 2015

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

1. Kapil Agrawal, Sadika Amreen, and Audris Mockus. 2015. *Commit quality in five high performance computing projects*. In Proceedings of the 2015 International Workshop on Software Engineering for High Performance Computing in Science, IEEE Press, pp. 24-29.
2. Iftekhar Ahmed, Soroush Ghorashi, and Carlos Jensen. 2014. *An Exploration of Code Quality in FOSS Projects*. OSS 2014, IFIP (International Federation for Information Processing), AICT 427, Corral, L.et al. (Eds.), pp. 181–190. Springer, Berlin, Heidelberg.
3. Oliver Arafat and Dirk Riehle. 2009. *The Commit Size Distribution of Open Source Software*. In Proc. HICSS'09 (Hawaii, USA, January 5-8, 2009). IEEE Computer Society Press, New York, NY, 2009, 1-8.
4. Amir Azarbakht and Carlos Jensen. 2014. *Drawing the Big Picture: Temporal Visualization of Dynamic Collaboration Graphs of OSS Software Forks*. OSS 2014, IFIP (IFIP International Federation for Information Processing) AICT 427, Corral L., *et al.* (Eds.).
5. Chris Beams. 2016. How to write a git commit message. http://chris.beams.io/posts/git-commit/[retrieved on 26 March 2016.
6. Evangelia Berdou. 2011. Organization in Open Source Communities: At the Crossroads of the Gift and Market Economies. Routledge.
7. Christian Bird and Nachiappan Nagappan . 2012. Who? Where? What? Examining Distributed Development in Two Large Open Source Projects. Proceedings of the 9th IEEE Working Conference on Mining Software Repositories, 237–246.
8. Tadeusz Chełkowski1, Peter Gloor, and Dariusz Jemielniak. 2016. Inequalities in Open Source Software Development: Analysis of Contributor's Commits in Apache Software Foundation Projects. PloS one 11, 4 (April, 2016).
9. D.J.Marcolesco.(retrieved on 28 July 2016).Writing good commit messages. https://github.com/erlang/otp/wiki/Writing-good-commit-messages
10. Paul A. David and Francesco Rullani. 2008. Dynamics of innovation in an "open source" collaboration environment: lurking, laboring, and launching FLOSS projects on SourceForge. Industrial and Corporate Change, 17(4) :647-710.
11. Amir Hossein Ghapanchi, Aybüke Aurum, and Farhad Daneshgar. 2012. The impact of process effectiveness on user interest in contributing to the open source software projects. Journal of software, 7(1): 212-219.
12. Jesus M. Gonzalez-Barahona, Gregorio Robles, Israel Herraiz, and Felipe Ortega. 2014. Studying the laws of software evolution in a long lived FLOSS project. Journal of Software: Evolution and Process, 26(7):589-612.
13. Carsten Kolassa, Dirk Riehle, and Michel Salim. 2013. The Empirical Commit Frequency Distribution of Open Source Projects. In: Proceedings of the 2013 joint International Symposium on Wikis and Open Collaboration, OpenSym'13, ACM.
14. Jérôme Kunegis, Sergej Sizov, Felix Schwagereit, and Damien Fay. 2012. Diversity dynamics in online networks. In: Proc. of the 23rd ACM Conf. on Hypertext and Social Media, USA.
15. Victor Kuechler, Claire Gilbertson, and Carlos Jensen. 2012. Gender Differences in Early Free and Open Source Software Joining Process. In: Hammouda, I., Lundell, B., Mikkonen, T., Scacchi, W. (eds.) OSS 2012. IFIP AICT, vol. 378, pp. 78–93. Springer, Heidelberg.
16. Tom Mens and Mathieu Goeminne .2011. Analysing the evolution of social aspects of open source software ecosystems. Eds: Jansen, Bosch, Ahmed, and Campell Proceedings of the Workshop on Software Ecosystems (IWSECO 2011).
17. Munish Saini and Kuljit Kaur. 2016. Change Profile Analysis of Open Source Software Systems to Understand their Evolutionary Behavior. Frontiers of Computer Science, Springer.
18. Eddie Santos and Abram Hindle. 2016. Judging a commit by its cover: correlating commit message entropy with build status on travis-CI. In Proceedings of the 13th International Conference on Mining Software Repositories (MSR '16). ACM, New York, NY, USA, 504-507.
19. G. Seber and Alan Lee. 2012. Linear regression analysis. Vol. 936. John Wiley & Sons.
20. Winters R. Scott. Score Normalization as a Fair Grading Practice. http://www.ericdigests.org/2003-4/score-normilization.html [retrieved on 20 July 2016].
21. Rodrigo Souza and Bruno Silva. 2017. Sentiment Analysis of Travis CI Builds. 2017. 14th International Conference on Mining Software Repositories.
22. M. R. Martinez Torres, S. L. Toral, M. Perales, and F. Barrero. 2011. Analysis of the Core Team Role in Open Source Communities. In: 2011 Int. Conf. on Complex, Intelligent and Software Intensive Systems (CISIS), pp. 109–114. IEEE.
23. Stanislav Levin and Amiram Yehudai. 2017. Boosting Automatic Commit Classification Into Maintenance Activities By Utilizing Source Code Changes. Pceedings of the 13th International Conference on Predictive Models and Data Analytics in Software Engineering Pages 97-106, Toronto, Canada — November 08 - 08, 2017

Thanks