

Six Easy Pieces

(of Quantitatively Analyzing Open Source Software)

Dirk Riehle

SAP Research, SAP Labs LLC

dirk@riehle.org, www.riehle.org, twitter.com/driehle



MARCH 24-25, 2009

Open Source Software

- Definition of open source software
 - Software that is **provided under an approved OSI license**
- General properties of open source software
 - Software that **is available in source code form**
 - Software that you **can modify and redistribute**
- Further important (not always present) properties
 - The project has gathered a thriving community
 - Feedback and ideas about the software are public and abundant
 - Projects are egalitarian, meritocratic, and self-organizing



Talk Overview (Agenda)

1. The Growth of Open Source Software
2. Data Mining for Fun and Profit
3. Efficiently Estimating Commit Sizes
4. The Commit Size Distribution of Open Source
5. Developer Activity in Open Source Software Projects
6. The Commenting Practice of Open Source
7. Team Size Evolution in Open Source Projects
8. Conclusions



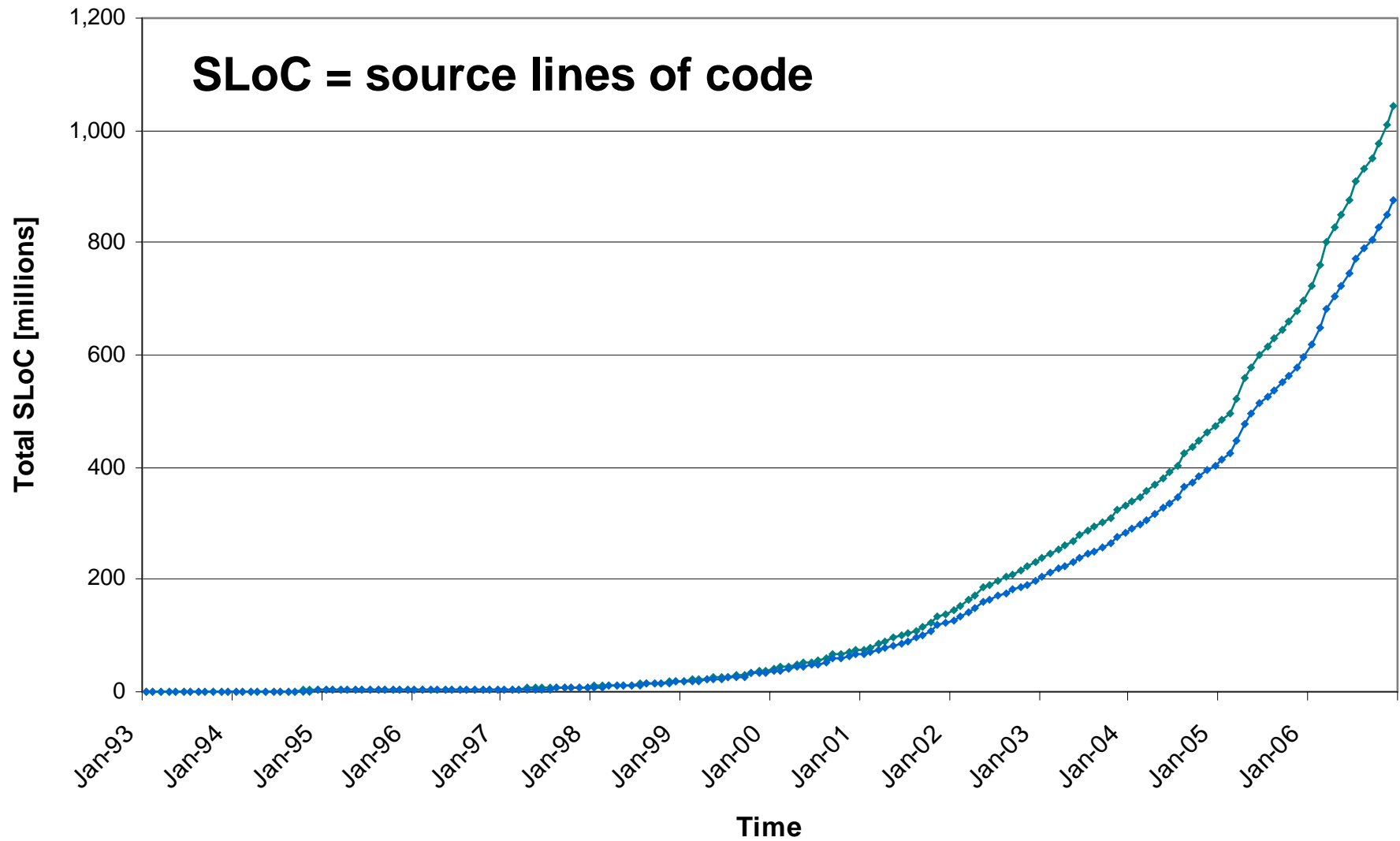
The Growth of Open Source Software

Amit Deshpande, Dirk Riehle. "The Total Growth of Open Source." In *Proceedings of the Fourth Conference on Open Source Systems (OSS 2008)*. Springer Verlag, 2008. Page 197-209.

<http://www.riehle.org/2008/03/14/the-total-growth-of-open-source/>



Source Code Growth in Open Source



Model of Source Code Growth

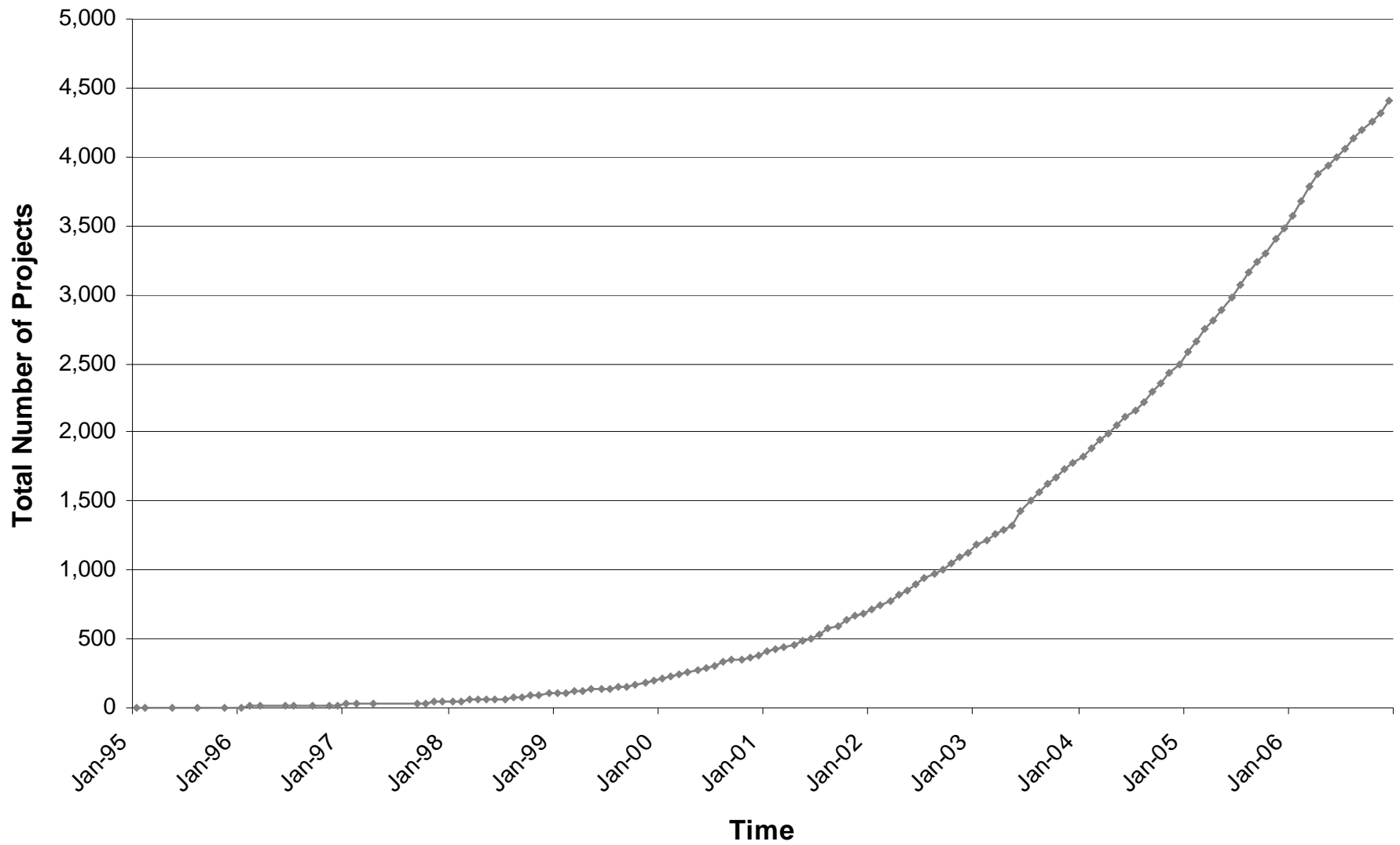
Approach	Model	R-square value
Upper bound	$y = 784098 * e^{0.0555x}$	0.961
Lower bound	$y = 2E+06 * e^{0.0464x}$	0.964

where,

y: Total open source lines of code

x: Time from Jan 1995 to Dec 2006 in months

Project Growth in Open Source



Model of Project Growth

Model	R-square value
$y = 7.1511e^{0.0499x}$	0.956
<p>where,</p> <p>y: Total number of open source projects</p> <p>x: Time from Jan 1995 to Dec 2006 in months</p>	

Where Open Source is Growing

It is not the size of individual projects,
but **the total number of** active working **open source**
projects that is **growing exponentially**

Open source is growing in every domain,
including **business applications**



Data Mining for Fun and Profit

Oliver Arafat, Amit Deshpande, Philipp Hofmann, Dirk Riehle.

<http://www.riehle.org/publications/>



MARCH 24-25, 2009

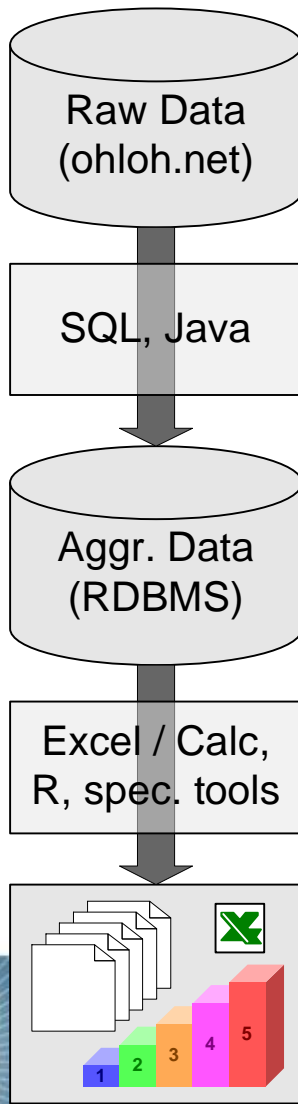
Motivation and Approach

- Gain insight into how open source software development works
 - Review processes unique to open source (or transferable)
 - Possibly improve corporate software development
 - Possibly build novel tools
- Post-facto detailed quantitative analysis of actual behavior
 - Analyzing what people do rather than what they say they do
 - Open source is publicly developed software so there is lots of data

Data Source, Data Quality

- Using the Ohloh.net repository of open source project information
 - Detailed information on a large number of projects (>9000 in 2008 snapshot)
 - Includes project structure and members but also detailed code information
 - Ohloh captures every commit down to diff sections (>8M commits)
 - See <http://www.ohloh.net> for more information
- To be useful, data needs to be cleaned and filtered
 - Depends on the question at hand
 - Developed a variety of easily applicable filters

Open Source Analytics Tool Chain



Raw data source

- Local database (ohloh.net snapshot, crawled sources)
- Web services access (ohloh.net, sourceforge.net, others)

Pre-processing

- Database querying using SQL and scripts
- Java library for computationally heavyweight filters, aggregation

Aggregated data source

- Output of pre-processing stage for specific analytical tasks
- Aggregated data significantly improves analysis speed

Analytical processing

- Mines aggregated (and raw) data for insights, hypothesis testing
- At present basic processing (Excel), machine learning next

Analysis output

- Results of analytical processing: averages, distributions, correlations
- Presented as models, tables, graphs, charts, etc.

Efficiently Estimating Commit Sizes

Philipp Hofmann, Dirk Riehle. “Estimating Commit Sizes Efficiently.” In *Proceedings of the 5th International Conference on Open Source Systems (OSS 2009)*. Springer Verlag, 2009. Forthcoming.

<http://www.riehle.org/2009/02/11/estimating-commit-sizes-efficiently/>



Definition of Commit Size

- Commit consists of Diffs which consist of Sections
 - One commit may affect multiple files, each file diff can have multiple sections
- Commit Size = $\text{sum}(\text{Diff Sizes})$ where Diff Size = $\text{sum}(\text{Section Sizes})$

What Diff Does

	a.txt	b.txt	diff a.txt b.txt
01:	a	a	4,5c4,6
02:	b	b	< d
03:	c	c	< f
04:	d	e	- - -
05:	f	e	> e
06:	g	e	> e
07:	h	g	> e
08:	m	h	7a9
09:	n	j	> j
10:		m	9d10
11:			< n



The Trouble with Diff

- From the GNU diff manual:

The way that GNU “diff” determines which lines have changed always comes up with a near-minimal set of differences. Usually it is good enough for practical purposes.

- Also, on the option “-d”:

Try hard to find a smaller set of changes.

- **There is no reliable way of determining SLoC changed!**

Some Diff Section Size Examples

Table 1: Interpretation of the entry 1 SLoC added, 1 SLoC removed

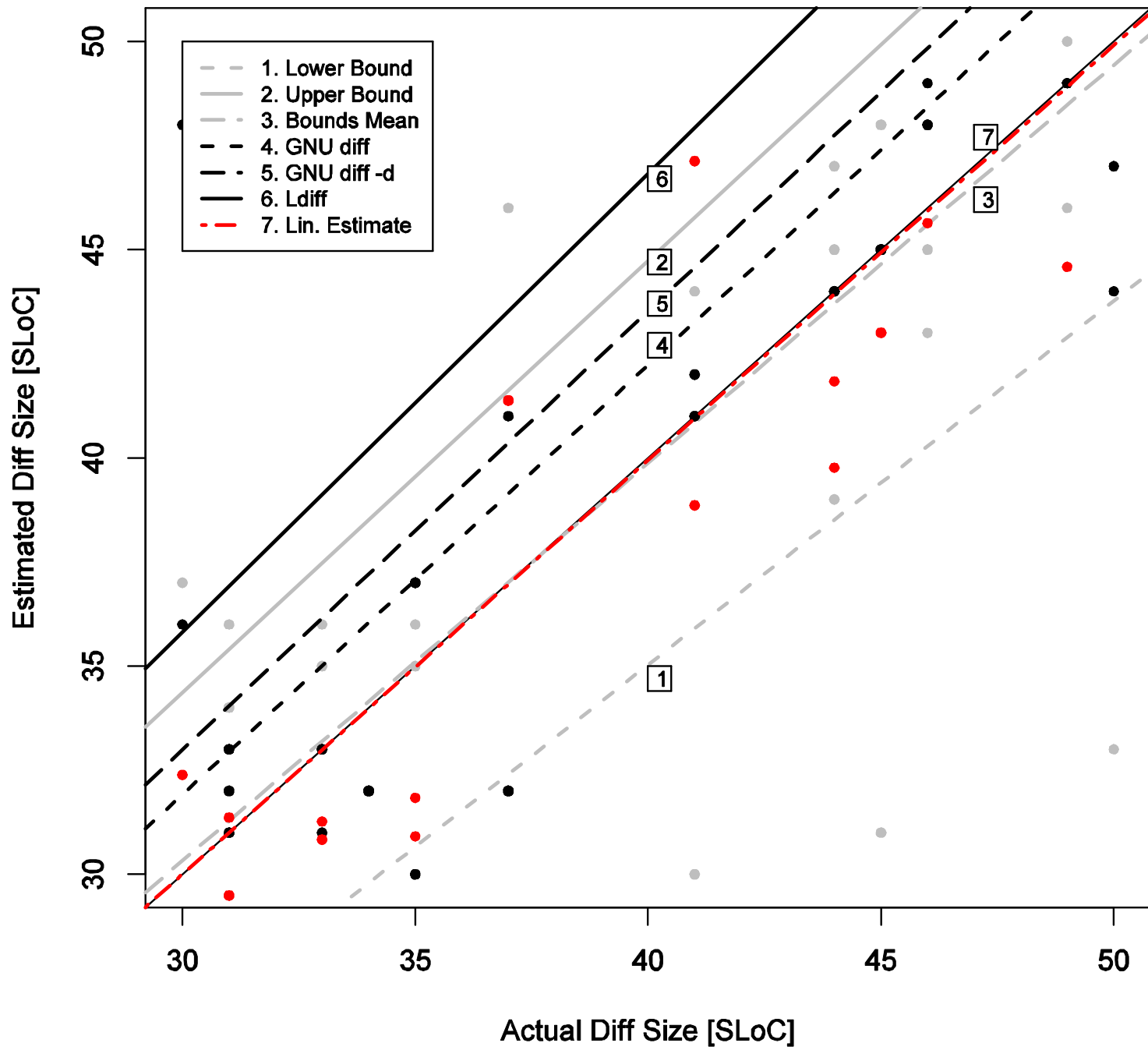
(1, 1)	Number of SLoC added	Number of SLoC removed	Number of SLoC changed	Number of Modifications
Event 1	0	0	1	1
Event 2	1	1	0	2

Table 2: Interpretation of the entry 4 SLoC added, 3 SLoC removed

(4, 3)	Number of SLoC added	Number of SLoC removed	Number of SLoC changed	Number of Modifications
Event 1	1	0	3	4
Event 2	2	1	2	5
Event 3	3	2	1	6
Event 4	4	3	0	7

Garden Variety of Heuristics

	Approach	Error Mean	Error Standard Deviation
1	Lower Bound	3.86	16.64
2	Upper Bound	-4.41	6.39
3	Bounds Mean	-0.27	7.68
4	GNU diff	-1.96	19.55
5	GNU diff -d	-3.06	30.87
6	Ldiff	-5.95	40.35
7	Linear Estimation	0	5.44



Definition of Commit Size

- General form: $\text{diff_size} \leftarrow (a, r)$
 - Diff size is a function of SLoC added and removed
- Linear form: $\text{diff_size}(a, r) = c_a \times a + c_r \times r + b$
 - Straightforward linear approximation
- Open source based estimation provides linear estimates
 - Linear regression run over open source sample data

```
function real diff_size(int a, int r)
    if (0.01269 × a + 0.01540 × r > 2.9965)
        return 0.9497 × a + 0.9744 × r - 2.9965
    else
        return 0.9370 × a + 0.9590 × r
    end
end
```

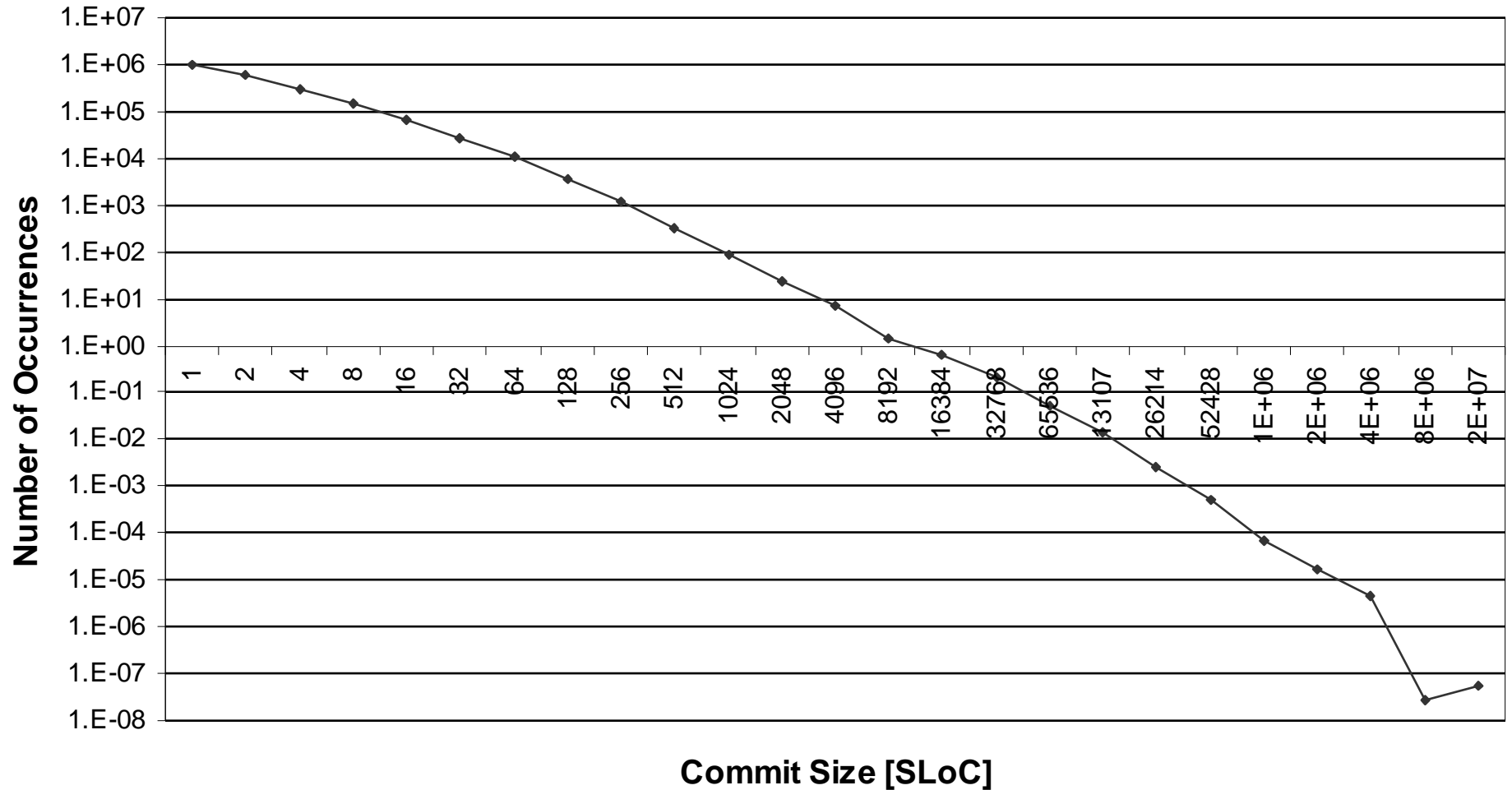
The Commit Size Distribution of Open Source

Oliver Arafat, Dirk Riehle. “The Commit Size Distribution of Open Source Software.” In *Proceedings of the 42nd Hawaiian International Conference on System Science (HICSS-42)*. IEEE Press: 2009. Page 1-8.

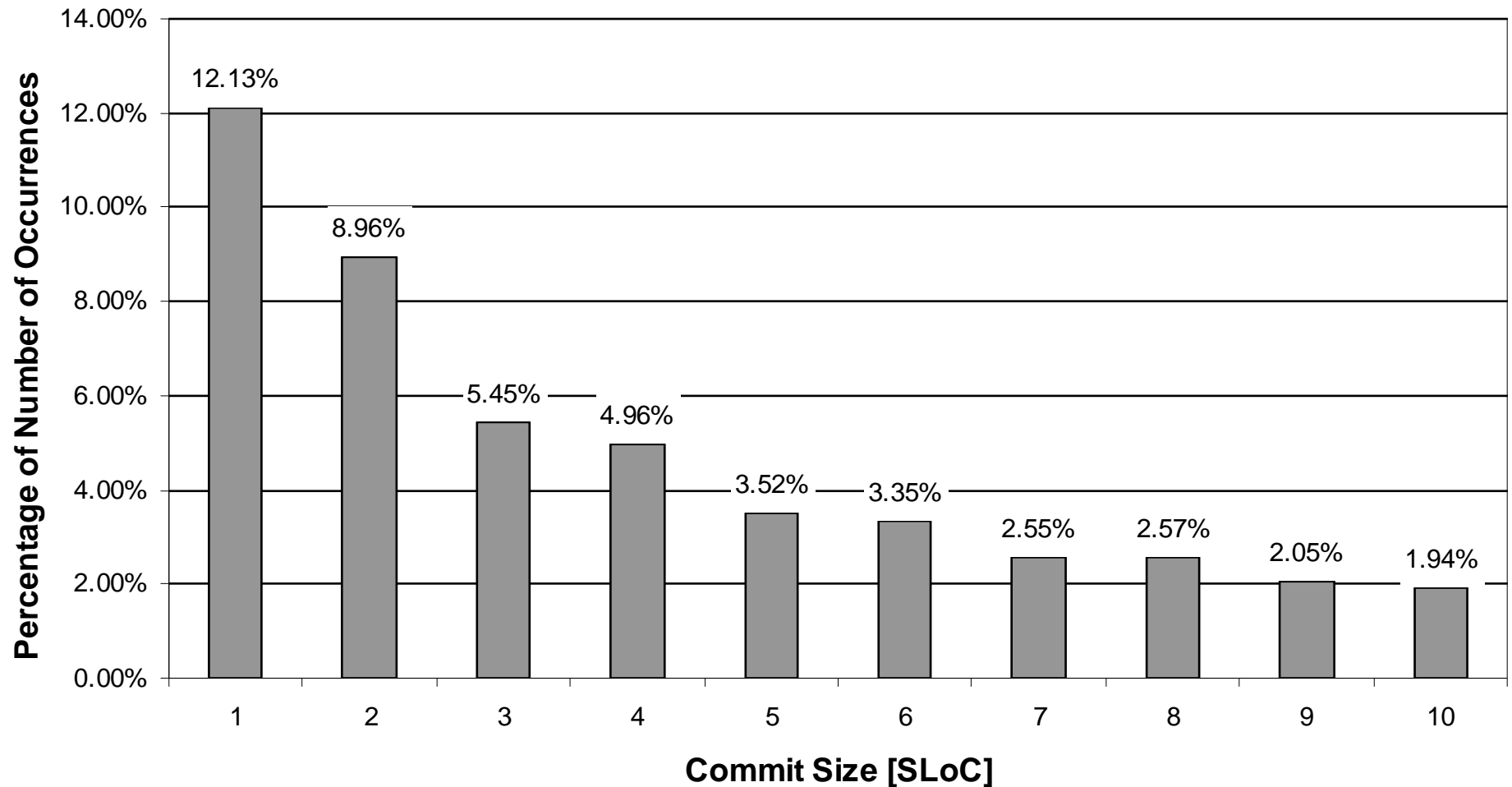
[http://www.riehle.org/2008/09/23/
the-commit-size-distribution-of-open-source-software/](http://www.riehle.org/2008/09/23/the-commit-size-distribution-of-open-source-software/)



The Overall Commit Size Distribution



The Dominance of Small Commits



The Overall Commit Size Distribution

Open source is incremental development

Small commits dominate: the smaller the commit, the more likely

Hypothesis: Contributors and committers follow same behavioral programming patterns

Given that our **development tools** were designed with 30-50 or more lines of code in mind, they **may be suboptimal** for open source

Further research into comparing open with closed source development is necessary



Developer Activity in Open Source Software Projects

Dirk Riehle, Oliver Arafat, Amit Deshpande. “Developer Activity in Open Source Software Projects.” In preparation.

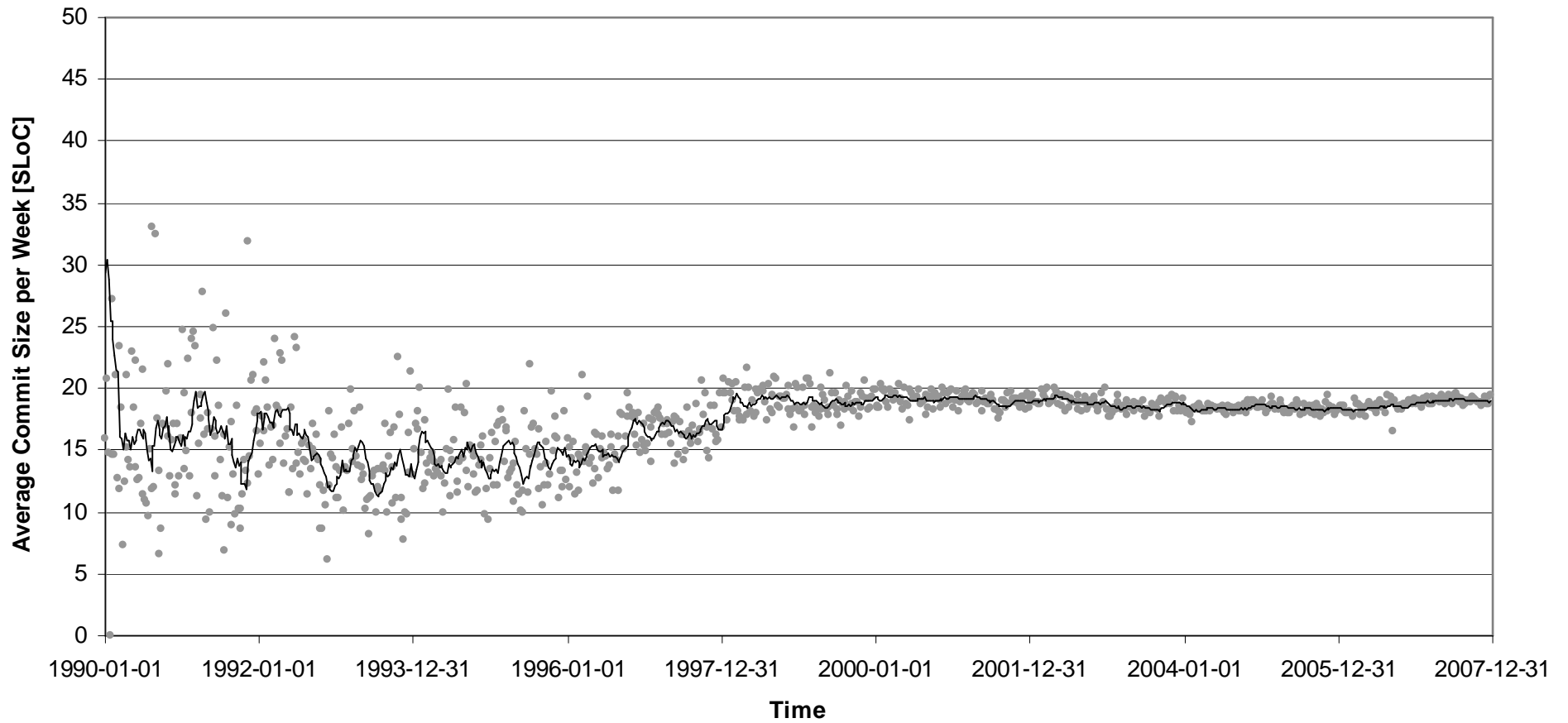
Amit Deshpande, Dirk Riehle. “Continuous Integration in Open Source Software Development.” In *Proceedings of the Fourth Conference on Open Source Systems (OSS 2008)*. Springer Verlag, 2008. Page 273-280.



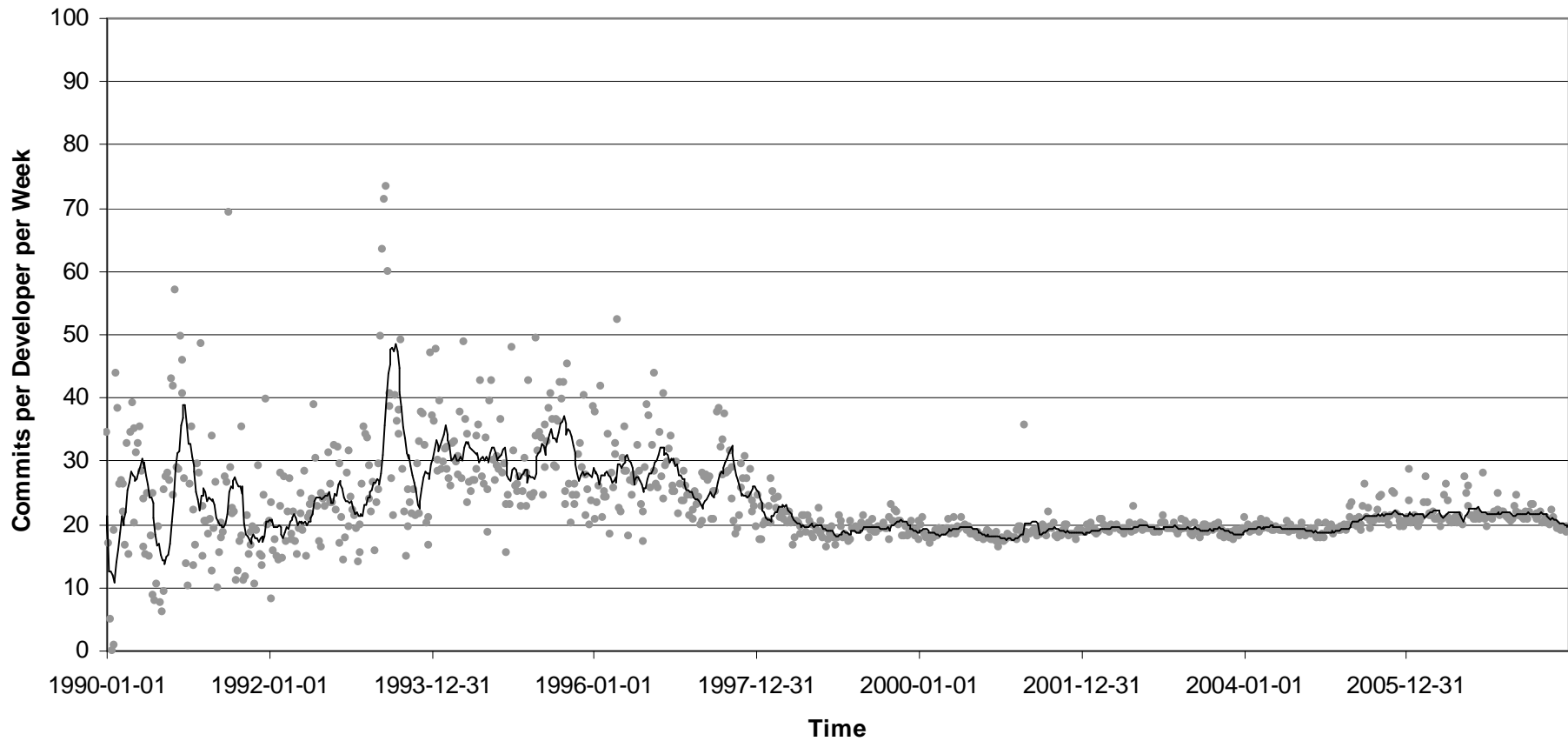
[http://www.riehle.org/2008/03/08/
continuous-integration-in-open-source-software-development/](http://www.riehle.org/2008/03/08/continuous-integration-in-open-source-software-development/)

MARCH 24-25, 2009

Average Commit Size



Average Commit Frequency



Changes in Developer Behavior

No significant changes apparent

Hypothesis: Foundations are not having an overall impact (yet)?

Agile methods, in particular **continuous integration**
did not change behavior or were **always present**

Further research into separating contributors from committers is
necessary

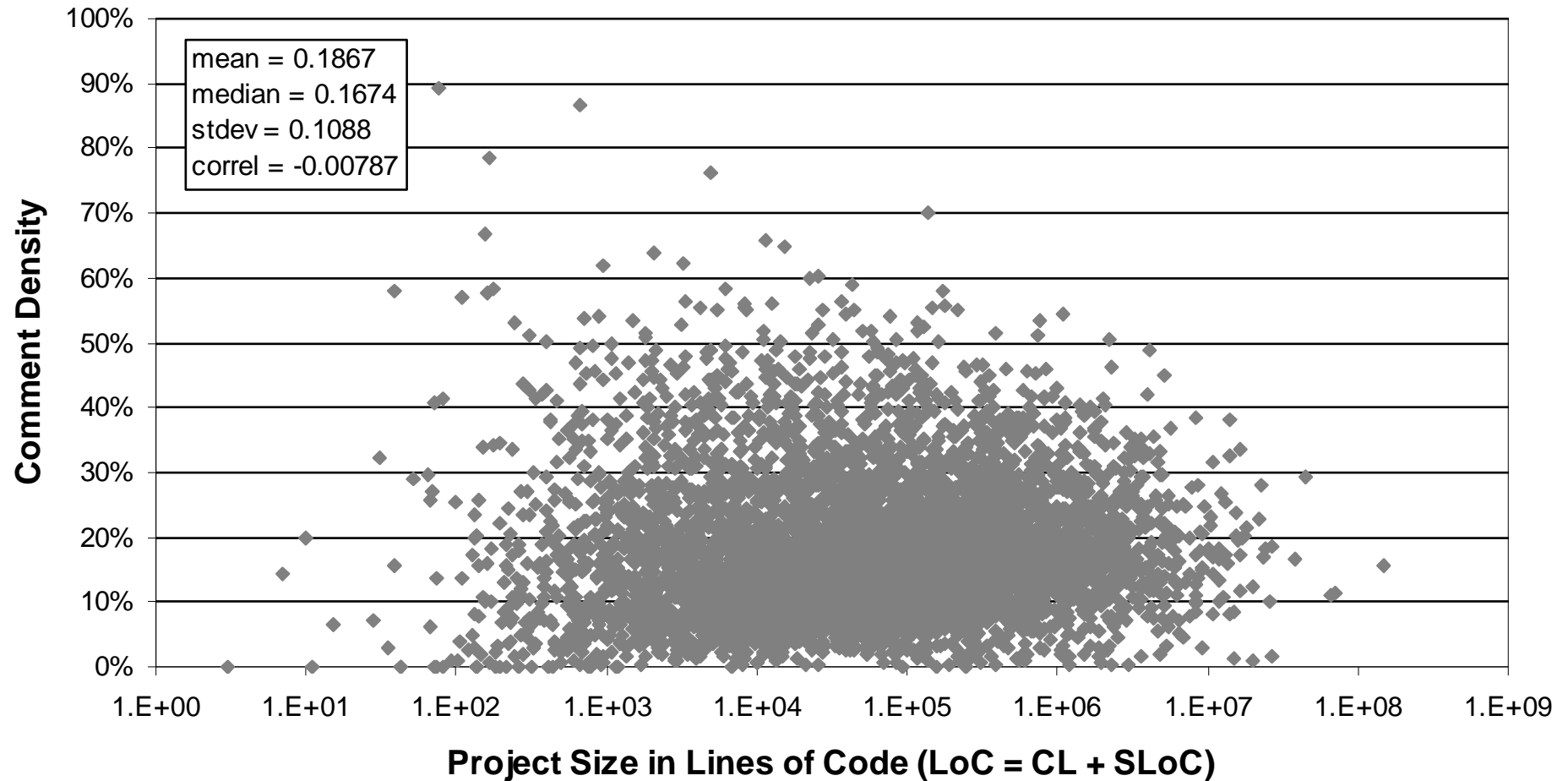
The Commenting Practice of Open Source

Oliver Arafat, Dirk Riehle. “The Comment Density of Open Source Software Code.” In *Companion to Proceedings of the 31st International Conference on Software Engineering (ICSE 2009)*. IEEE Press, 2009: Forthcoming.

[http://www.riehle.org/2009/02/04/
the-comment-density-of-open-source-software-code/](http://www.riehle.org/2009/02/04/the-comment-density-of-open-source-software-code/)



Average Comment Density

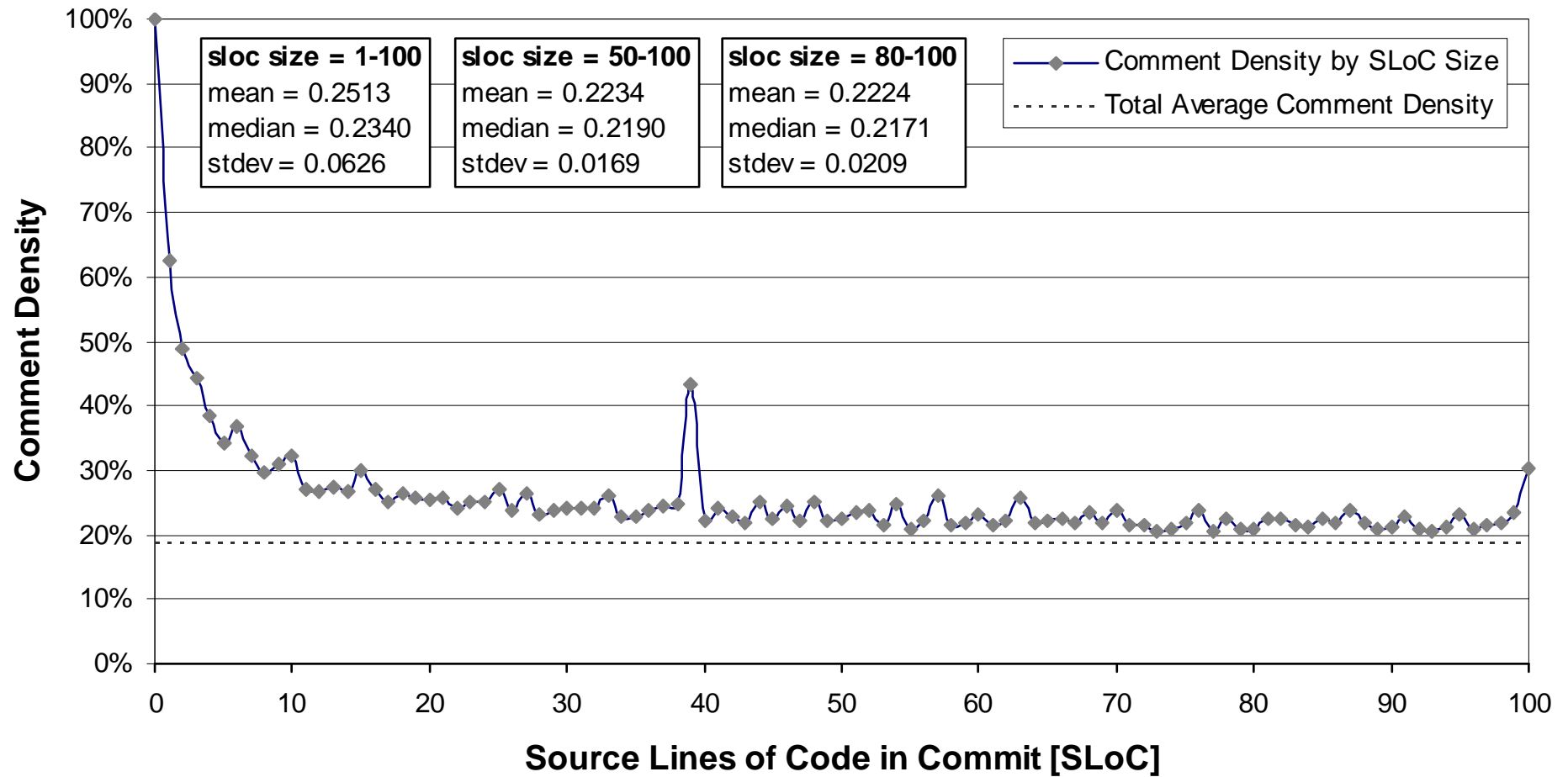


comment density = #comment lines / #code lines

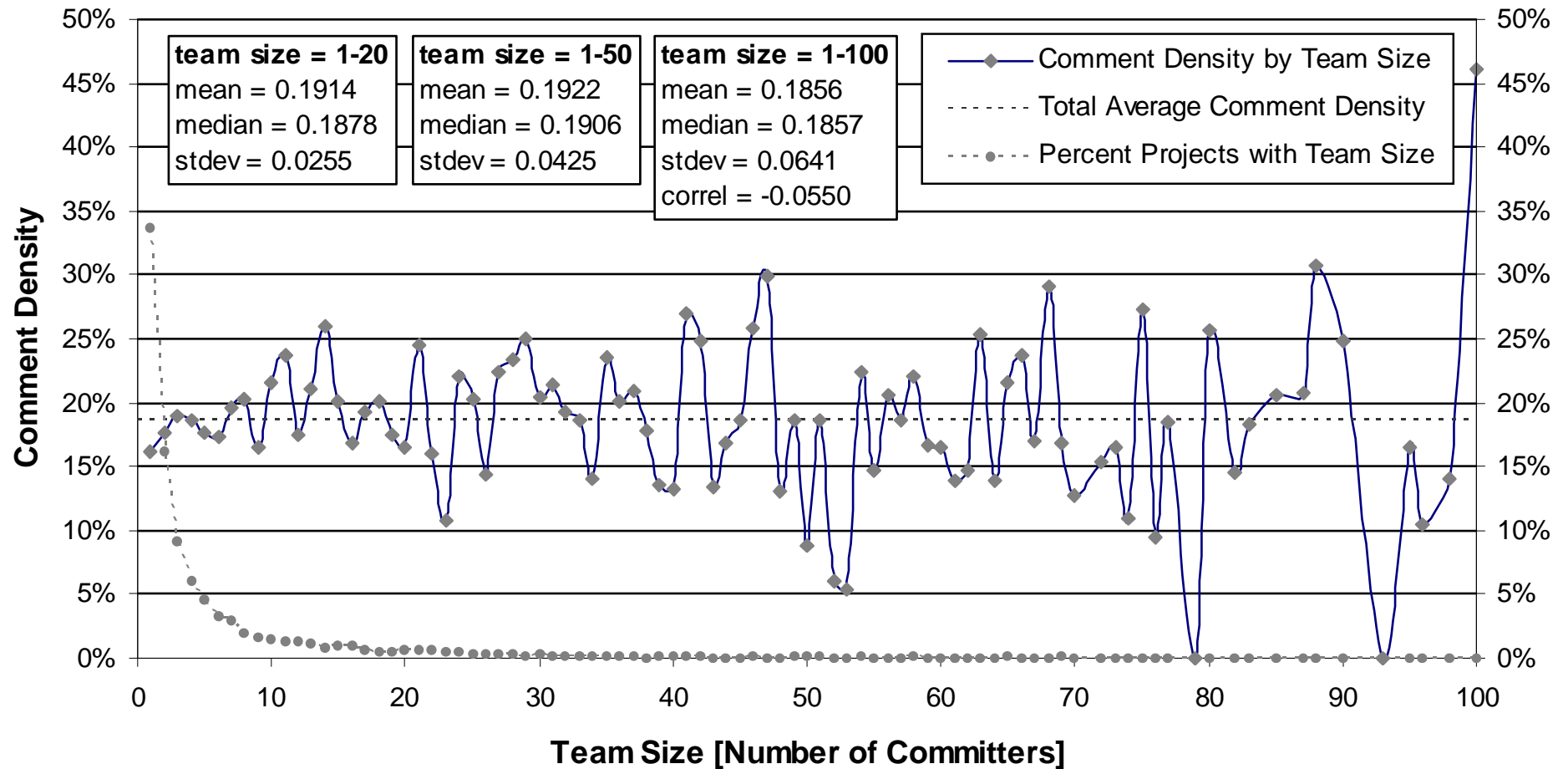
Comment Density by Programming Language

#	Language	Average [%]	Stddev [%]	Population Size
1.	Java	26%	11%	1085
2.	php	22%	12%	559
3.	C/C++	18%	8%	1621
4.	Javascript	16%	9%	276
5.	Python	11%	8%	534
6.	Perl	10%	7%	273

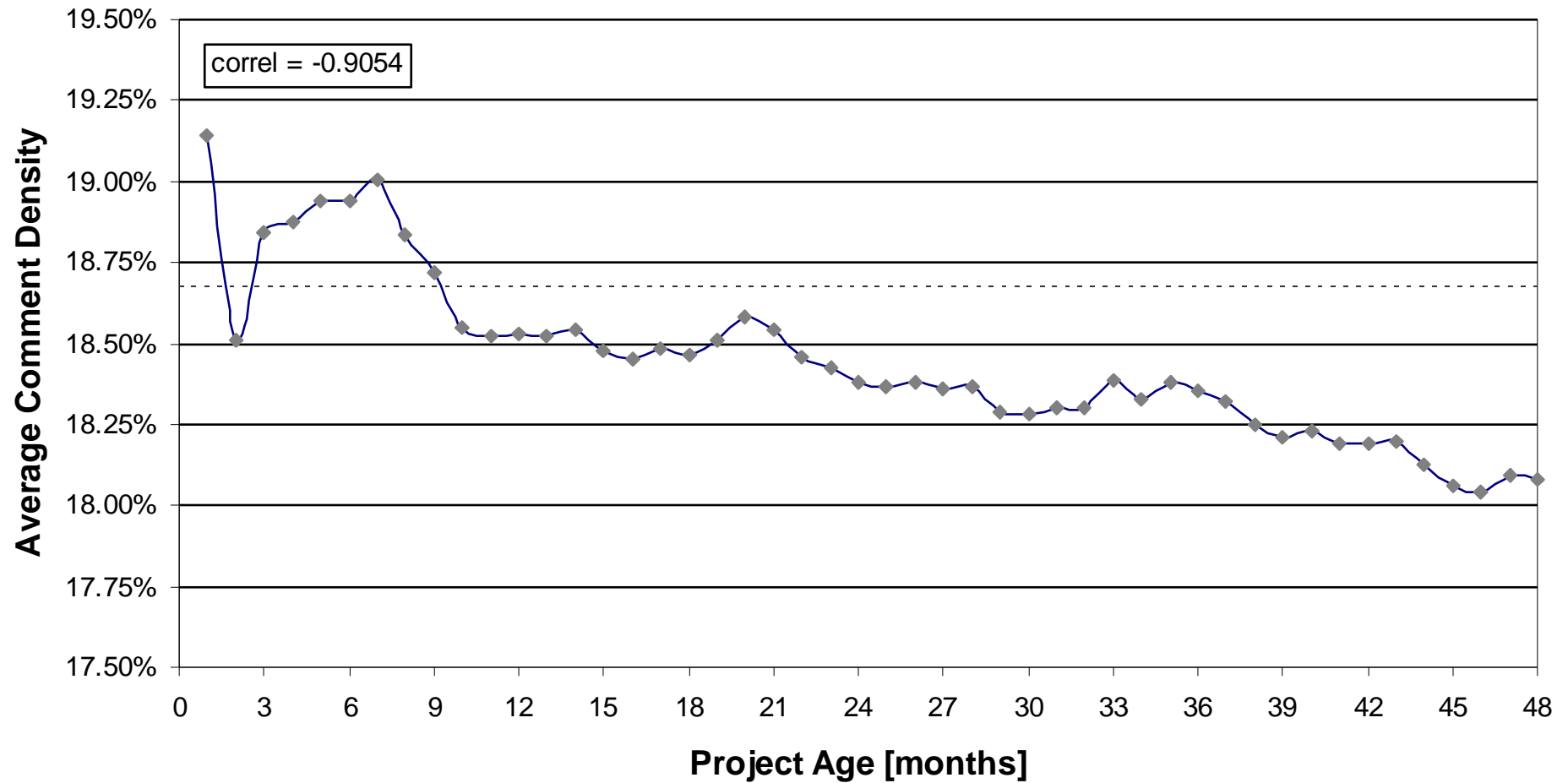
Comment Density by Commit Size



Comment Density by Team Size



Comment Density by Project Age



Commenting in Open Source

A continuous on-going practice

Surprisingly high comment density,
clearly **no belief in self-documenting code**

Hypothesis: **Comment density** of **open source**
represents **the sweet spot** of commenting code

For any specific conclusions, watch the parameters

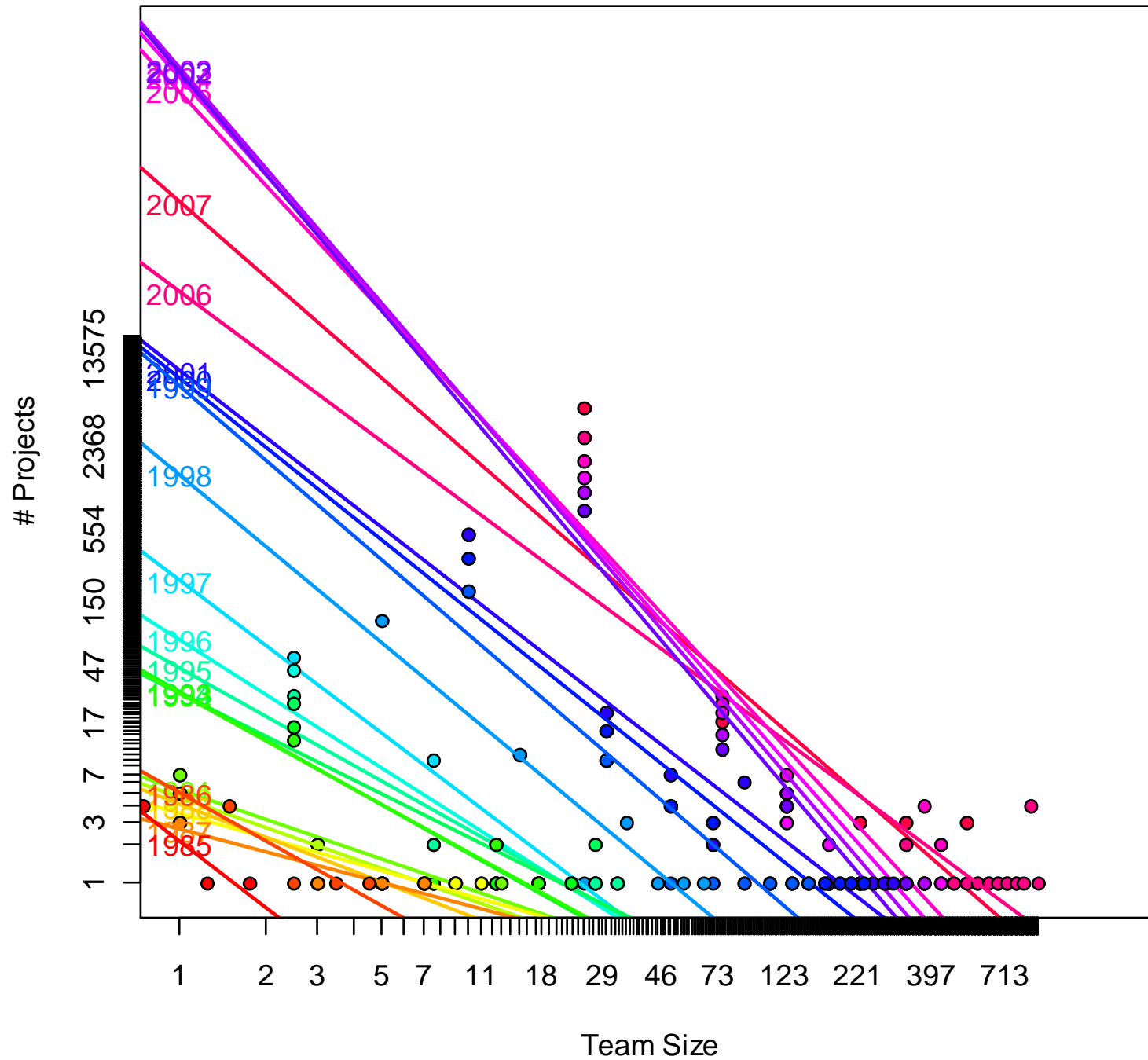
Further research into project types is necessary



Team Size Evolution in Open Source Projects

Philipp Hofmann, Dirk Riehle. "Team Size Evolution in Open Source Software Projects." In preparation.





Is Open Source Scale-Free?

- Definition of scale-free distribution
 - Follows power-law (straight line on log-log graph)
 - Very common in nature, technology, and society
 - Implies same principles govern every level of scale
- Examples of scale-free graphs in open source
 - Commit size distribution
 - Team size distribution
- If open source was scale-free then
 - We would know how to scale to ever larger project sizes
 - By basically applying the same principles across the board

Conclusions

- Lots of interesting insights to be gained from analyzing open source
- We have the chance now to fix misconceptions by looking at data
- Lots of that insight may apply to closed source development as well
- Open source may show the most resource-efficient sweet spots
- New research agenda: Is open source scale-free?

Thank you!

dirk@riehle.org, www.riehle.org, twitter.com/driehle

Comments are welcome!



MARCH 24-25, 2009