Software Analytics
Harlan D. Mills Award Acceptance Speech

Nachi Nagappan
About Me

- My name is Nachiappan. I also go by Nachi.

- [https://nachinagappan.github.io/](https://nachinagappan.github.io/)

- Graduated with a PhD with Laurie Williams.

- I read a lot of Franco-Belgian comics (Bande dessinées)

- Attend Comic conventions

- Miniature railroad modeling (HO and G).
What metrics are the **best predictors of failures**?

What is the **data quality** level used in empirical studies and how much does it actually matter?

I just submitted a **bug report**.
Will it be fixed?

How can I tell if a piece of software will have **vulnerabilities**?

Do **cross-cutting concerns** cause defects?

Does **Test Driven Development (TDD)** produce better code in shorter time?

If I increase **test coverage**, will that actually increase software quality?

Are there any **metrics that are indicators of failures** in both Open Source and Commercial domains?

Should I be writing **unit tests** in my software project?

Is strong **code ownership** good or bad for software quality?

Does **Distributed/Global software development** affect quality?

What is the **data quality** level used in empirical studies and how much does it actually matter?

I just submitted a **bug report**.
Will it be fixed?

How can I tell if a piece of software will have **vulnerabilities**?

Do **cross-cutting concerns** cause defects?

Does **Test Driven Development (TDD)** produce better code in shorter time?

If I increase **test coverage**, will that actually increase software quality?

Are there any **metrics that are indicators of failures** in both Open Source and Commercial domains?

Should I be writing **unit tests** in my software project?

Is strong **code ownership** good or bad for software quality?

Does **Distributed/Global software development** affect quality?
History of Software Analytics

1971: Fumio Akiyama
first known “size” law
(Defects ~ LOC)

1976: Thomas McCabe
code complexity

1981: Barry Boehm
effort estimation
History of Software Analytics

Empirical Software Engineering
Experimental Software Engineering

1980
Victor Basili

1990s
Dieter Rombach

2000s
Audris Mockus
Elaine Weyuker
Thomas Ostrand
and many others
History of Software Analytics

Mining Software Repositories

2004

Ahmed Hassan
History of Software Analytics

Mining Software Repositories
The PROMISE Repository/Conference
History of Software Analytics

Nachi Nagappan founds Empirical Software Engineering at Microsoft.

DJ Patil and Jeff Hammerbacher coin the term “data scientist” to define their jobs at LinkedIn and Facebook.

2005

2007

2008

2009

2010

Ahmed Hassan founds the SAIL Group in Canada.

Dongmei Zhang founds the Software Analytics Group at Microsoft Research Asia.

NSF Workshop on Future of SE
Functional languages rack up best scores for software quality

Taking short breaks during training can help you improve more quickly, video game study finds

Free Apps With Ads May Be Killing Your Phone's Battery And Data Plan

Older Computer Programmers Not Out of Touch, Study Finds
Data Scientist: The Sexiest Job of the 21st Century

by Thomas H. Davenport and D.J. Patil

FROM THE OCTOBER 2012 ISSUE

When Jonathan Goldman arrived for work in June 2006 at LinkedIn, the business networking site, the place still felt like a start-up. The company had just under 8 million accounts, and the number was growing quickly as existing members invited their friends and colleagues to join. But users weren’t seeking out connections with the people who were already on the site at the rate executives had expected. Something was apparently missing in the social experience. As one LinkedIn manager put it, “It was like arriving at a conference reception and realizing you don’t know anyone. So you just stand in the corner sipping your drink—and you probably leave early.”

Goldman, a PhD in physics from Stanford, was intrigued by the linking he did see going on and by the richness of the user profiles. It all made for messy data and unwieldy analysis, but as he began exploring people’s connections, he started to see possibilities. He began forming theories, testing hunches, and finding patterns that allowed him to predict whose networks a given profile would land in. He could imagine that new features capitalizing on the heuristics he was developing might...
Obsessing over our customers is everybody's job. I'm looking to the engineering teams to **build the experiences our customers love.** [...] In order to deliver the experiences our customers need for the mobile-first and cloud-first world, we will modernize our engineering processes to be **customer-obsessed, data-driven, speed-oriented and quality-focused.**

http://news.microsoft.com/ceo/bold-ambition/index.html
Each engineering group will have **Data and Applied Science resources** that will focus on measurable outcomes for our products and predictive analysis of market trends, which will allow us to innovate more effectively.

http://news.microsoft.com/ceo/bold-ambition/index.html
Looking back…
Dr. Thomas Ball

Static analysis tools as early indicators of pre-release defect density.  
ICSE 2005

Use of relative code churn measures to predict system defect density.  
ICSE 2005

Assessing the Relationship between Software Assertions and Faults: An Empirical Investigation.  
ISSRE 2006
Dr. Andreas Zeller

Mining metrics to predict component failures. ICSE 2006

Extrinsic influence factors in software reliability: a study of 200,000 windows machines. ICSE 2014
Dr. Prem Devanbu


Putting It All Together: Using Socio-technical Networks to Predict Failures. ISSRE 2009.

Don't touch my code!: examining the effects of ownership on software quality. FSE 2011.
Dr. Victor Basili

The influence of organizational structure on software quality: an empirical case study. ICSE 2008
Dr. Harald Gall

Cross-project defect prediction: a large scale experiment on data vs. domain vs. process. **ESEC/SIGSOFT FSE 2009**

Does distributed development affect software quality? An empirical case study of Windows Vista. **ICSE 2009**

Software engineering for machine learning: a case study. **ICSE (SEIP) 2019**: 291-300
Dr. Miryung Kim

A field study of refactoring challenges and benefits. SIGSOFT FSE 2012
Dr. Emerson Murphy-Hill

The design of bug fixes. ICSE 2013: 332-341

Cowboys, ankle sprains, and keepers of quality: how is video game development different from software development? ICSE 2014
Dr. David Lo

How practitioners perceive the relevance of software engineering research. ESEC/SIGSOFT FSE 2015


Dr. Audris Mockus

Test coverage and post-verification defects: A multiple case study. **ESEM 2009**
The Future

NEXT EXIT
SAINT

Software Analysis and Intelligence
The 3 P’s of Productivity

- People
- Product
- Process
# SAINT Focus Areas

<table>
<thead>
<tr>
<th>Developer Communities</th>
<th>Future of Software Creation</th>
</tr>
</thead>
<tbody>
<tr>
<td>• GitHub, Visual Studio</td>
<td>• Understanding how developers and processes work now and identifying future trends</td>
</tr>
<tr>
<td>• Fostering empathy in developer communities</td>
<td>• Software 2.0</td>
</tr>
<tr>
<td>• Understanding how identity-based signals support an inclusive, open environment</td>
<td>• Productivity of software teams</td>
</tr>
<tr>
<td>• Non-traditional software engineer experience</td>
<td>• Remote work</td>
</tr>
</tbody>
</table>

The common goal of our work is to understand productivity and build interventions to better support programmers.
SAINT Focus Areas – Milestones

Developer Communities
- PR Acceptance Bias
- Gender Diversity in GitHub projects

Future of Software Creation
- Effort estimation at the PR level
- SE for ML
DEVELOPER COMMUNITIES
More specifically the goal of this study

To understand the influence of geographical location on pull request acceptance decisions in GitHub?

- Geographical location of submitters
- Same geographical location of submitters and integrators
The Data Source

- GHTorrent data
- 1069 projects and 370,411 pull requests developed in Python (357), Java (315), Ruby (359), and Scala (38).
- Represent top 1% of the projects developed by using pull requests as the mode of collaboration.
- We use countryNameManager script by Bogdan et al and others from UC Davis.
Observations

• Controlling for the confounding effects of
  – Project characteristics
  – Developer characteristics
  – Pull request characteristics

Geographical location explains significant differences in pull request acceptance decisions.
Observations

✔ Compared to the United States, submitters from United Kingdom (22%), Canada (25%), Japan (40%), Netherlands (43%), and Switzerland (58%) have higher chances of getting their pull requests accepted.

✔ However, submitters from Germany (15%), Brazil (17%), China (24%), and Italy (19%) have lower chances of getting their pull requests accepted compared to the United States.
Observations

✓ Submitters and integrators having the same nationality increases the chances of pull request acceptance decisions by 19% compared to when submitters and integrators are from different countries.
Observations

Submitters

✓ Submitters from some countries perceive to experience bias more compared to other countries.

✓ Observations in agreement with quantitative analysis.
Observations

Integrators

✓ 53% more integrators perceive that they encourage submitters from their nationality to participate.

✓ 8 out of 10 integrators feel that it is easy to work with submitters from the same nationality.

✓ Integrators do not feel that submitters from some nationalities are better at writing pull requests compared to others, except for India.
Gender Diversity in GitHub

- Worked with a large international collaboration between MSR, SMU, DELFT, IIIT.

<table>
<thead>
<tr>
<th>Region Level 1</th>
<th>Region Level 2</th>
<th>Count</th>
<th>Man</th>
<th>Woman</th>
<th>Unknown</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>Northern Africa</td>
<td>91</td>
<td>91.21</td>
<td>5.49</td>
<td>3.33</td>
</tr>
<tr>
<td>Africa</td>
<td>Sub-Saharan Africa</td>
<td>273</td>
<td>92.67</td>
<td>3.66</td>
<td>3.66</td>
</tr>
<tr>
<td>Americas</td>
<td>Latin America and the Caribbean</td>
<td>2547</td>
<td>93.29</td>
<td>4.75</td>
<td>1.96</td>
</tr>
<tr>
<td>Americas</td>
<td>Northern America</td>
<td>24055</td>
<td>90.27</td>
<td>7.47</td>
<td>2.25</td>
</tr>
<tr>
<td>Americas</td>
<td>Others</td>
<td>5</td>
<td>80.00</td>
<td>0.00</td>
<td>20.00</td>
</tr>
<tr>
<td>Asia</td>
<td>Central Asia</td>
<td>34</td>
<td>88.24</td>
<td>2.94</td>
<td>8.82</td>
</tr>
<tr>
<td>Asia</td>
<td>Eastern Asia</td>
<td>2585</td>
<td>80.46</td>
<td>10.10</td>
<td>9.44</td>
</tr>
<tr>
<td>Asia</td>
<td>South-eastern Asia</td>
<td>686</td>
<td>87.90</td>
<td>6.85</td>
<td>5.25</td>
</tr>
<tr>
<td>Asia</td>
<td>Southern Asia</td>
<td>1463</td>
<td>91.46</td>
<td>5.47</td>
<td>3.08</td>
</tr>
<tr>
<td>Asia</td>
<td>Western Asia</td>
<td>529</td>
<td>93.19</td>
<td>3.40</td>
<td>3.40</td>
</tr>
<tr>
<td>Europe</td>
<td>Eastern Europe</td>
<td>3858</td>
<td>94.35</td>
<td>2.90</td>
<td>2.75</td>
</tr>
<tr>
<td>Europe</td>
<td>Northern Europe</td>
<td>7541</td>
<td>92.71</td>
<td>5.38</td>
<td>1.91</td>
</tr>
<tr>
<td>Europe</td>
<td>Southern Europe</td>
<td>2314</td>
<td>94.77</td>
<td>3.11</td>
<td>2.12</td>
</tr>
<tr>
<td>Europe</td>
<td>Western Europe</td>
<td>10637</td>
<td>92.94</td>
<td>3.88</td>
<td>3.18</td>
</tr>
<tr>
<td>Oceania</td>
<td>Australia and New Zealand</td>
<td>1870</td>
<td>92.62</td>
<td>5.13</td>
<td>2.25</td>
</tr>
<tr>
<td>Oceania</td>
<td>Melanesia</td>
<td>5</td>
<td>80.00</td>
<td>0.00</td>
<td>20.00</td>
</tr>
<tr>
<td>Oceania</td>
<td>Polynesia</td>
<td>5</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Unknown</td>
<td>Unknown</td>
<td>12123</td>
<td>61.96</td>
<td>6.22</td>
<td>31.82</td>
</tr>
</tbody>
</table>
Results

• There is no strong correlation between gender and geographic diversity.
• Since 2014, there has been a small and statistically significant improvement of gender diversity in North America and South-Eastern Asia, but negligible change elsewhere.
Results

• Many of the barriers and motivations for contributing converge across geographic region.
  – Lack of resources
  – Goal alignment shift
  – Poor engineering environment
  – Poor working environment
  – Unclear onboarding
  – Inactivity on projects
FUTURE OF SOFTWARE CREATION
Application areas

“Please briefly describe your AI-based product, feature, or service in a few sentences.”

<table>
<thead>
<tr>
<th>Ads</th>
<th>Human Resources</th>
<th>Business</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Legal</td>
<td>Content Moderation</td>
</tr>
<tr>
<td></td>
<td>Mobile</td>
<td>Customers</td>
</tr>
<tr>
<td></td>
<td>Office</td>
<td>Devices</td>
</tr>
<tr>
<td></td>
<td>Research</td>
<td>Drawing</td>
</tr>
<tr>
<td></td>
<td>Security</td>
<td>Education</td>
</tr>
<tr>
<td></td>
<td>Windows</td>
<td>Environment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gaming</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Healthcare</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Incident Management</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Infrastructure</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Knowledge Graph</td>
</tr>
<tr>
<td></td>
<td></td>
<td>News</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Software Engineering</td>
</tr>
<tr>
<td></td>
<td></td>
<td>User activity / UX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VR/AR</td>
</tr>
</tbody>
</table>
Algorithms in use

Classification
Clustering
Dynamic Programming
Signal Processing
Statistics

Fraud Detection
Navigation
Knowledge Graph
Root Cause Analysis
Social Network Analysis
User Behavior Modeling
Tools/Services

Search
Relevance
Ranking
Query Understanding

Recommendation
Profile Matching
Collaborative Filtering
Visual Design

Prediction
Risk Prediction
Forecasting
(Sales and Marketing)

Decision Optimization
Resource Optimization
Planning
Pricing
Bidding
Process Optimization

Vision
Face Recognition
Gesture Recognition
Image Understanding
OCR

Speech
Speech-to-Text
Speaker Identification

NLP
Entity Recognition
Q&A
Sentiment Analysis
Bots
Intent Prediction
Summarization
Machine Translation
Grammar Checking
Ontology Construction
Text Similarity
Machine Learning Workflow

Average reported hours/week

4.4 | 4.7 | 4.5 | 2.9 | 4.6 | 5.4 | 3.8 | 5.1 | 2.6

Percent of respondents who work on this activity

44% | 42% | 40% | 30% | 38% | 40% | 40% | 36% | 29%
Common challenges

End-to-end tool fragmentation
Tools can make ML too difficult
Data collection and cleaning is hard
Education and lack of expertise
Debugging is hard
Model evaluation and deployment
Challenges differ by experience

Common to Everyone

- End-to-end tool fragmentation
- Data collection, cleaning, management

Low experience

- Education and training
- Integrating AI into larger systems

High experience

- Need for specific tools
- Scalability
- Educating others
- Model evolution, evaluation, and deployment
Best practices for machine learning

ML tools need to be better stitched into the ML workflow and the workflow needs to be automated.

Center development around data (sharing, provenance, versioning).

Educating non-specialists in ML takes a lot of time but it worth the effort. Leverage internal training and knowledge sharing.

ML models are difficult to debug. Using simple, explainable, and composable models helps.

Use carefully designed test sets, score cards for evaluating combo flights, and human-in-the-loop evaluation.

Do not decouple model building from the rest of the software.
Data, data, data

Traditional software engineering focuses mainly on code, not on data.

– How is data stored, versioned, and tracked in repositories?
– Data must be changed out every few months to satisfy compliance requirements.

Teams suggest

– “Pay a lot of attention to the data.”
– “Put more effort on data collection and annotation”
– “Be relaxed about framework / machine learning code, but careful & deliberate about data & objectives.”
– “Standardize on terminology and naming conventions such as the same type of user_id”
– “Reuse the modules or data as much as possible to reduce duplicate effort.”
Effort Estimation

“In software development, *effort estimation* is the process of predicting the most realistic amount of effort (expressed in terms of person-hours or money) required to develop or maintain software based on incomplete, uncertain and noisy input.”
Overruns

66% of enterprise software projects have cost overruns

22% of enterprise software projects go beyond estimated schedule

17% of IT projects go so bad that they can threaten the very existence of the company

Based on McKinsey report
## Model

### Process
- Number of active PRs at this time, Is it a bug fix, Is it a feature, Number of reviewers

### Developer
- Age of the developer in current team, Age of the developer in current repo, Age of the developer in Microsoft

### Churn
- Class churn, Method churn, Loop churn, Class member churn, Loc changed

### Temporal
- Day of the week, Average age of PRs With similar paths

### Architectural
- Feature Space
  - Number of paths touched, Number of distinct file types, Is csproj being edited, Is it a refactor, Is it a deprecate
## Feature Correlations

<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Day of the week</td>
<td>• Age of the developer in current repo</td>
</tr>
<tr>
<td>• Average age of PRs of a developer</td>
<td>• Age of the developer in current team</td>
</tr>
<tr>
<td>• Average age of PRs which touched similar paths</td>
<td>• Is the PR fixing a bug?</td>
</tr>
</tbody>
</table>
Nudge Comment

**Sankie Service**  04/01/2019

Analyzing historic data and trends, PRs like this tend to be completed in 110 hours (approx). As it is already 2 days past the estimated time frame, you may want to consider driving this PR towards completion.

Please provide feedback/comments/questions [here](#).

This data was generated by machine learning suggestions. Please do one of the following:

a) Please Resolve the comment if the comment is reasonable.

b) Please mark the comment as Won't Fix if it is not relevant.

---

**Derek**  04/02/2019

This was reasonable. This PR sat stale while doing work for FHL week, so it was untouched for an extended period.

Write a reply...
Evaluation

<table>
<thead>
<tr>
<th>Category</th>
<th>Count</th>
<th>Time from Decoration - Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Notified PRs</td>
<td>1655</td>
<td>103.07</td>
</tr>
<tr>
<td>Notified PRs</td>
<td>1069</td>
<td>71.27</td>
</tr>
</tbody>
</table>

No. of PRs completed after notifying

Average time to completion

PRs are moving faster by 44%
Anecdotes

“The pipeline is failing and blocking this check in. Followed up with an ICM incident!”

“I thought the approximation was pretty good. Making few more changes and pushing this PR through! Thanks!”

“The approximation does sound about right. I went on-call which led to delay in check-in in this case. Normally, it would have been within about that range.”

Comment resolution percentage is 73.3%
Isaac Newton in 1675: "If I have seen further it is by standing on the shoulders of Giants."
THANKS TO...
Thank you!