The Sound of Data

Thomas Zimmermann, Microsoft Research

the sound of music (1965)

The Sound of Music

The story of Maria von Trapp, a "young Austrian woman studying to become a nun in Salzburg in 1938 who is sent to the villa of a retired naval officer and widower to be governess to his seven children. After bringing love and music into the lives of the family through kindness and patience, she marries the officer and together with the children they find a way to survive the loss of their homeland through courage and faith." (Wikipedia.org)

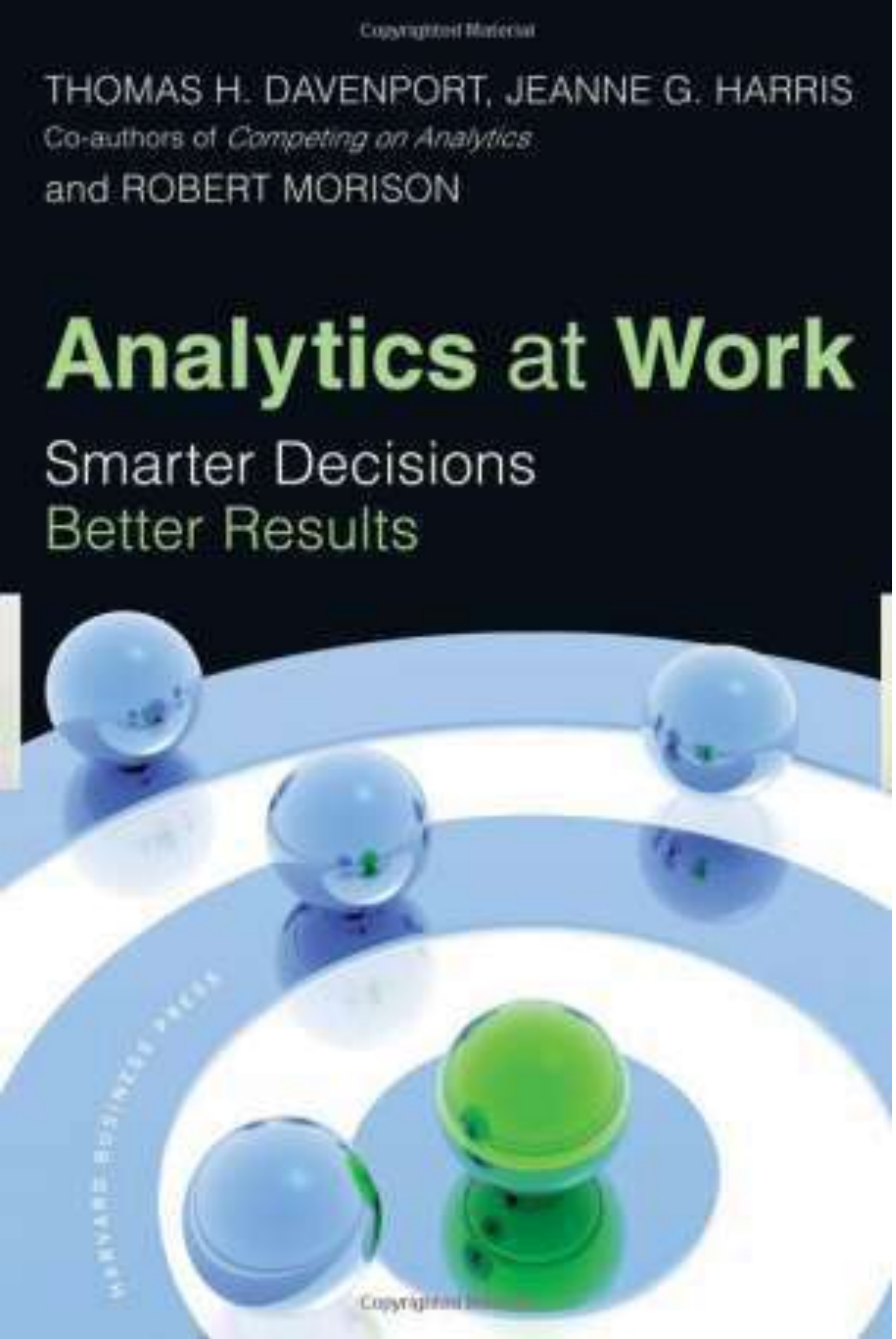
Passionate about music.
Excites other people about music

Huge success:
book, German films, Broadway musical, Hollywood film

Happy ending

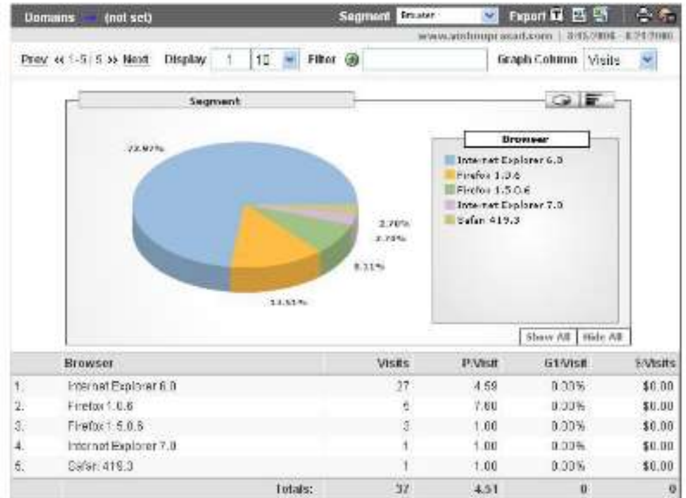
The Sound of Music	The Sound of Data
The story of Maria von Trapp, a “young Austrian woman studying to become a nun in Salzburg in 1938 who is sent to the villa of a retired naval officer and widower to be governess to his seven children. After bringing love and music into the lives of the family through kindness and patience, she marries the officer and together with the children they find a way to survive the loss of their homeland through courage and faith.” (Wikipedia.org)	The story of data scientists who are sent to society to bring love and insight into the lives of people.
Passionate about music Excites other people about music	Passionate about data Excite other people about data
Huge success: book, German films, Broadway musical, Hollywood film	Huge success: books, films, Moneyball no Broadway musical (yet)
Happy ending	Happy ending?

data science / analytics 101



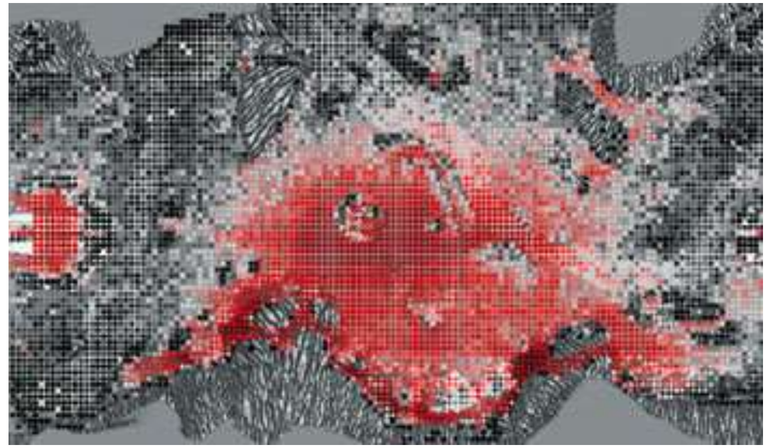
Use of data, analysis, and systematic reasoning to [inform and] make decisions

web analytics



(Slide by Ray Buse)

game analytics

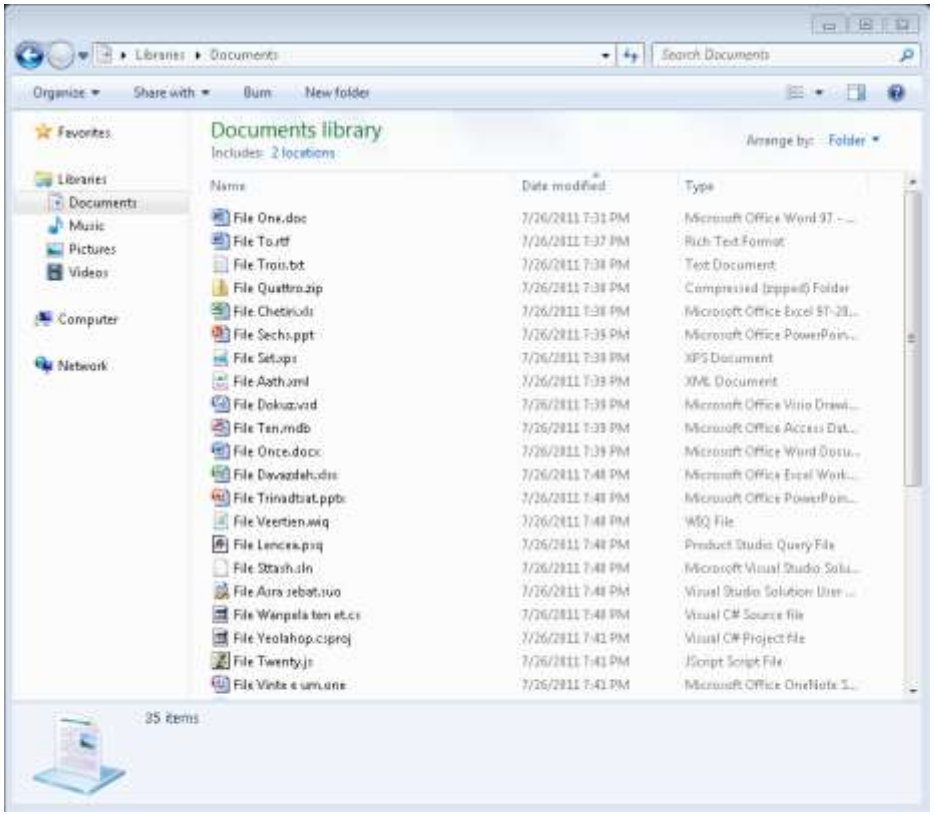


Halo heat maps

Free to play

usage analytics

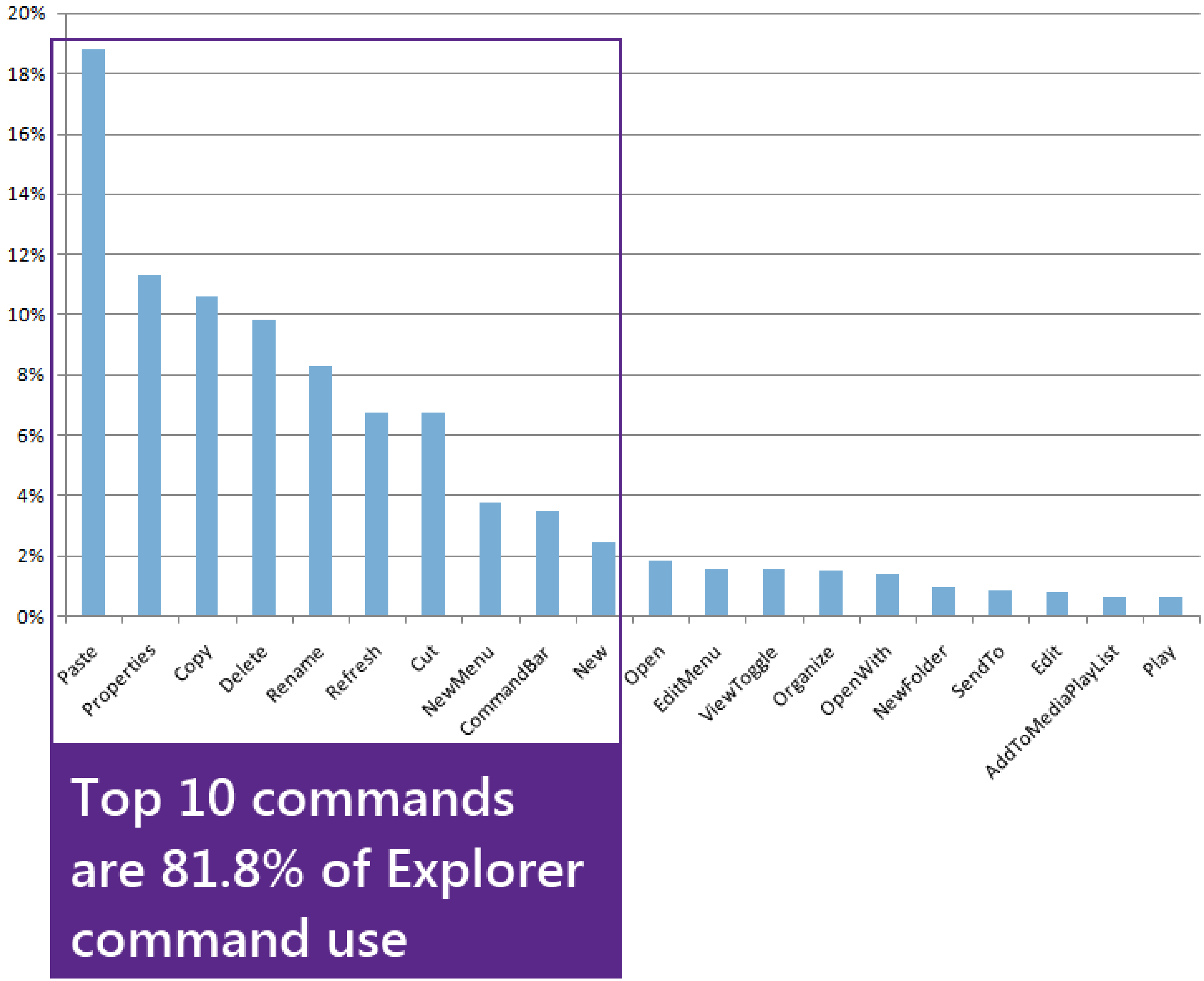
Improving the File Explorer for Windows 8

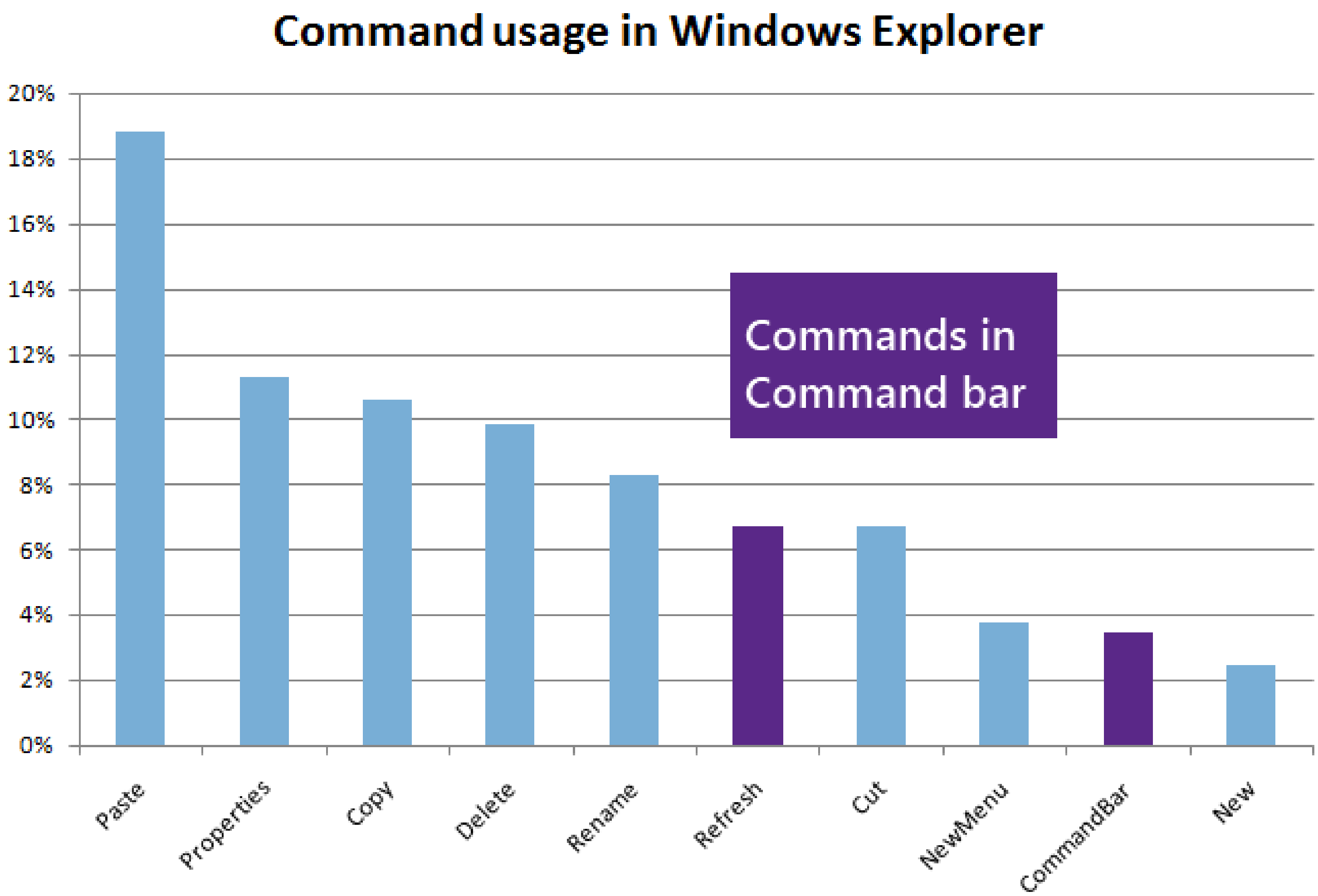


Explorer in Windows 7

Alex Simons: Improvements in Windows Explorer.
http://blogs.msdn.com/b/b8/archive/2011/08/29/improvements-in-windows-explorer.aspx

Command usage in Windows Explorer

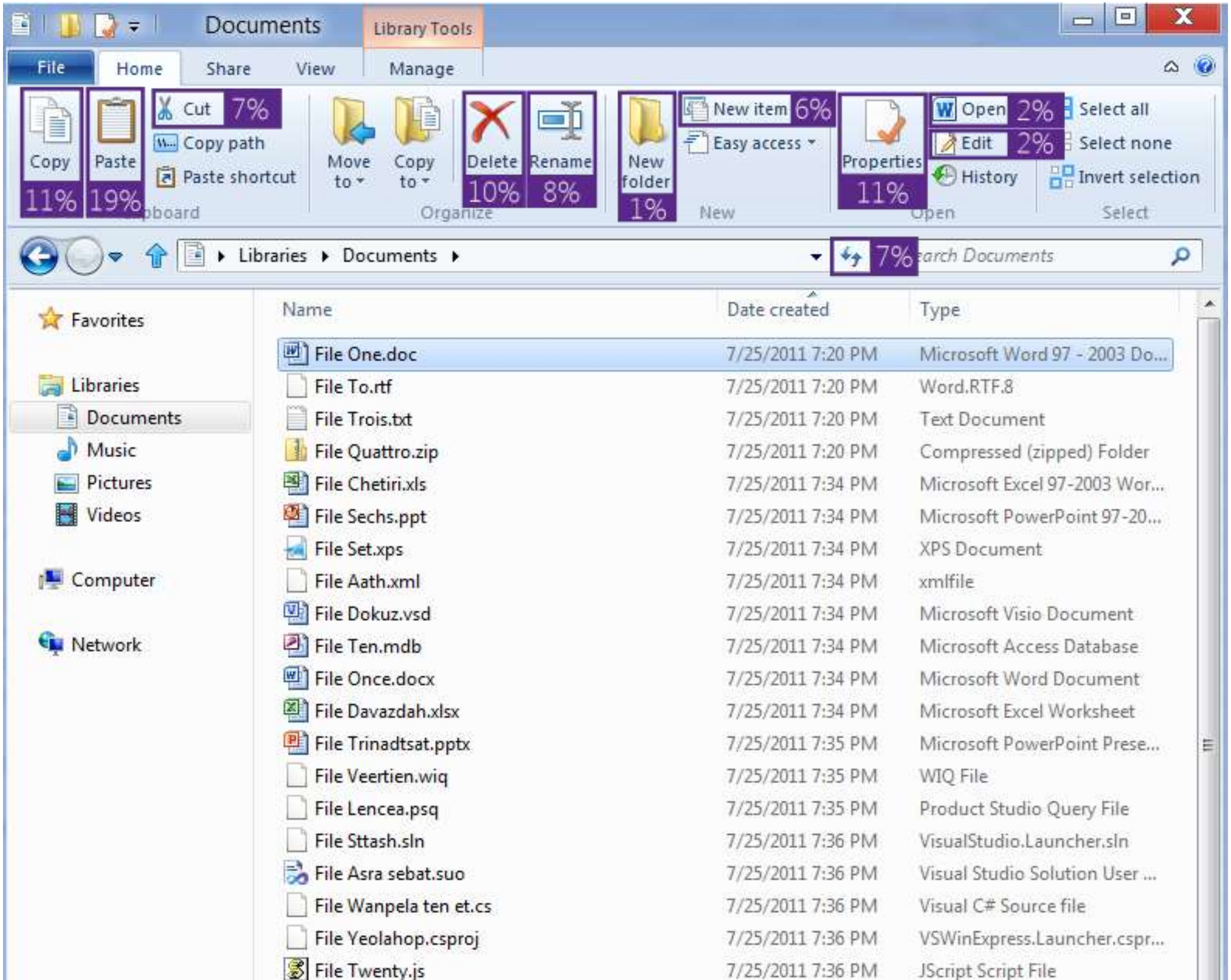



Command usage in Windows Explorer
20%
18%
16%
14%
12%
10%
8%
6%
4%
2%
0%
Commands in Command bar
Paste
Properties
Copy
Delete
Rename
Refresh
Cut
NewMenu
CommandBar
New

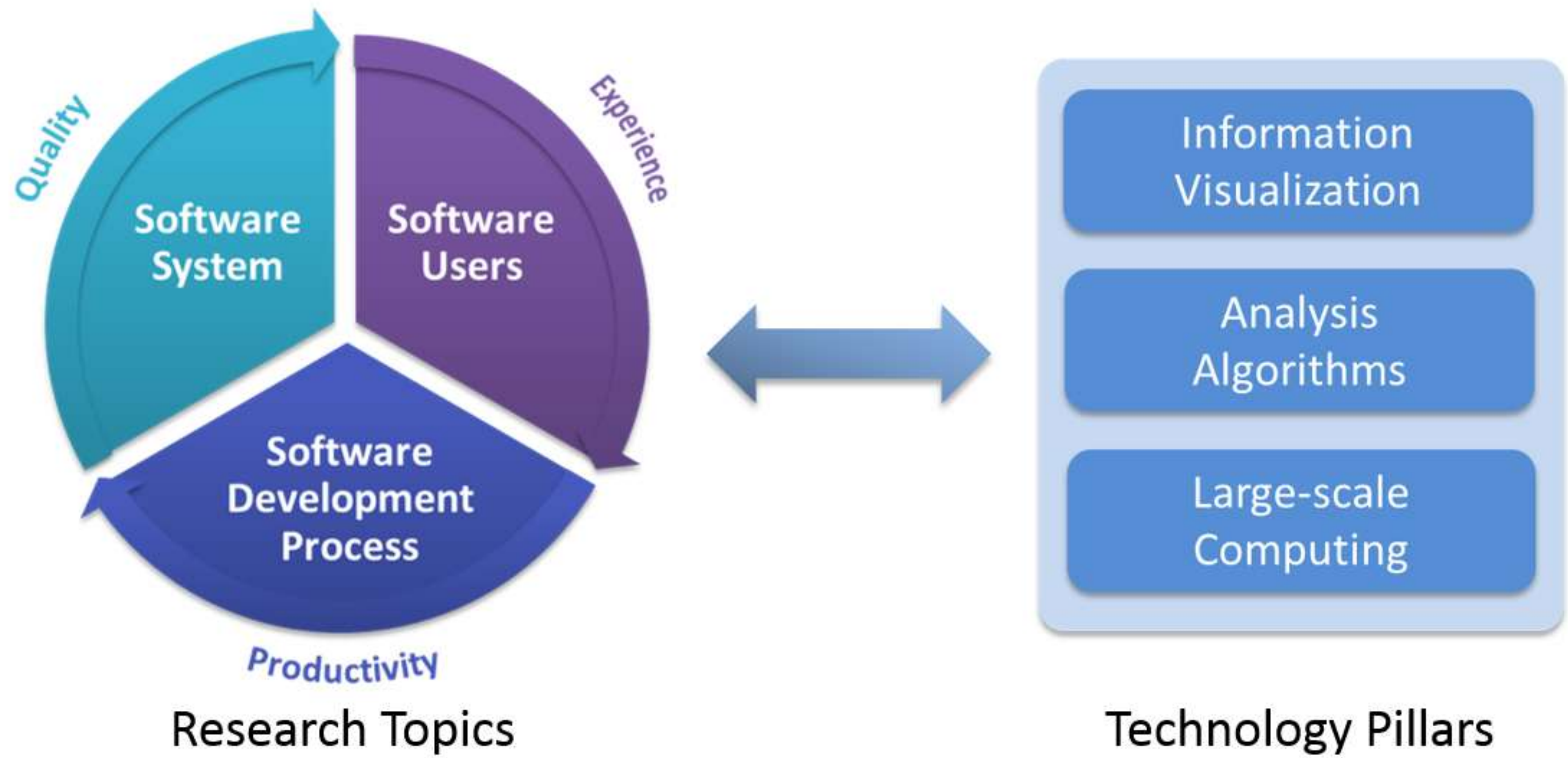
Customer feedback

- Bring back the "Up" button from Windows XP,
- Add cut, copy, & paste into the top-level UI,
- More customizable command surface, and
- More keyboard shortcuts.

Overlay showing Command usage % by button on the new Home tab



trinity of software analytics



Dongmei Zhang, Shi Han, Yingnong Dang, Jian-Guang Lou, Haidong Zhang, Tao Xie: Software Analytics in Practice. IEEE Software 30(5): 30-37, September/October 2013.

MSR Asia Software Analytics group: http://research.microsoft.com/en-us/groups/sa/

history of software analytics

EARLY "GLOBAL" MODELS AND SOFTWARE ANALYTICS

As soon as people started programming, it became apparent that programming was an inherently buggy process. As recalled by Maurice Wilkes,[1] speaking of his programming experiences from the early 1950s: "It was on one of my journeys between the EDSAC room and the punching equipment that 'hesitating at the angles of stairs' the realization came over me with full force that a good part of the remainder of my life was going to be spent in finding errors in my own programs."

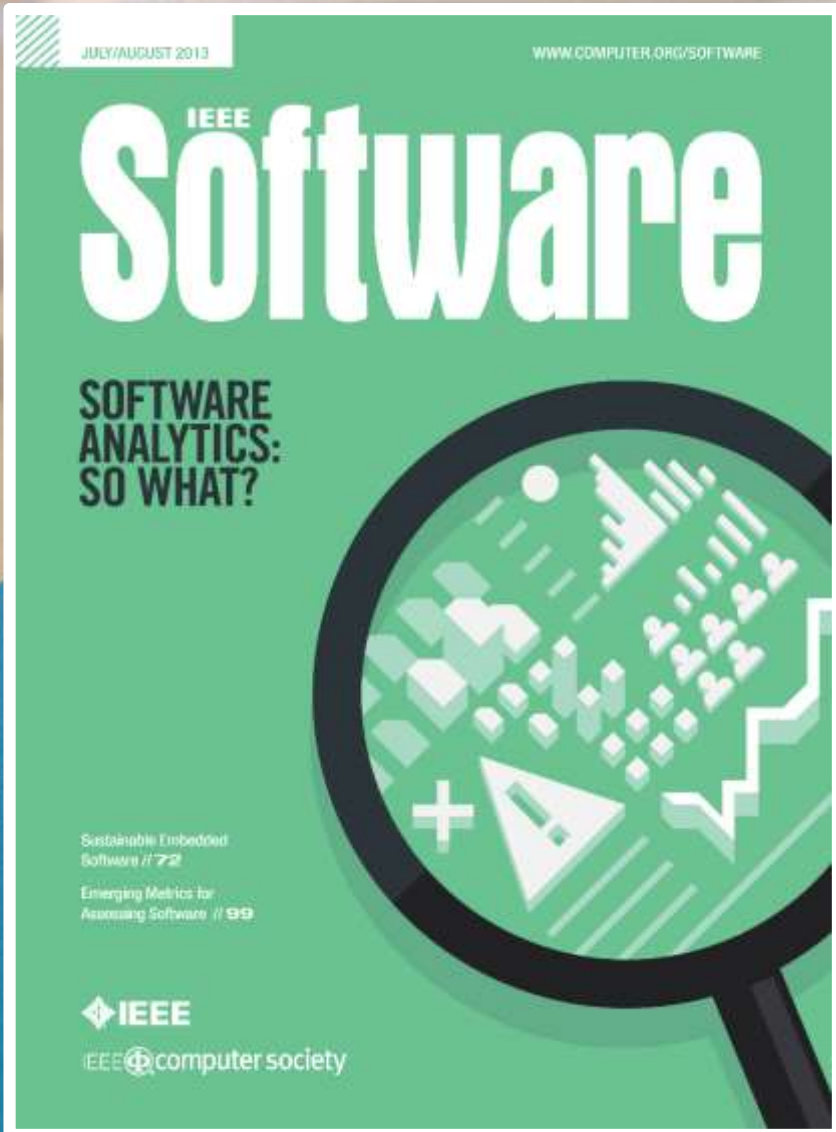
It took several decades to gather the experience required to quantify the size/defect relationship. In 1971, Fumio Akiyama[2] described the first known "size" law, saying the number of defects D was a function of the number of LOC; specifically, $D = 4.86 + 0.018 * i$. In 1976, Thomas McCabe argued that the number of LOC was less important than the complexity of that code.[3] He argued that code is more likely to be defective when his "cyclomatic complexity" measure was over 10.

Not only is programming an inherently buggy process, it's also inherently difficult. Based on data from 63 projects, Barry Boehm[4] proposed in 1981 an estimator for development effort that was exponential on program size: effort = a * KLOCb * EffortMultipliers, where $2.4 \le a \le 3$ and $1.05 \le b \le 1.2$.

References

1. M. Wilkes, *Memoirs of a Computer Pioneer*, MIT Press, 1985.
2. F. Akiyama, "An Example of Software System Debugging," *Information Processing*, vol. 71, 1971, pp. 353–359.
3. T. McCabe, "A Complexity Measure," *IEEE Trans. Software Eng.*, vol. 2, no. 4, 1976, pp. 308–320.
4. B. Boehm, *Software Engineering Economics*, Prentice-Hall, 1981.

Tim Menzies, Thomas Zimmermann: Software Analytics: So What?
IEEE Software 30(4): 31-37 (2013)


JULY/AUGUST 2013
WWW.COMPUTER.ORG/SOFTWARE
IEEE Software
SOFTWARE ANALYTICS: SO WHAT?
Sustainable Embedded Software // 72
Emerging Metrics for Assessing Software // 99
IEEE
IEEE computer society


SEPTEMBER/OCTOBER 2013
WWW.COMPUTER.ORG/SOFTWARE
IEEE Software
THE MANY FACES OF SOFTWARE ANALYTICS
IEEE
IEEE computer society

Contributors: Alberto Bacchelli, Olga Baysal, Ayse Bener, Aditya Budi, Bora Caglayan, Gul Calikli, Joshua Charles Campbell, Jacek Czerwonka, Kostadin Damevski, Madeline Diep, Robert Dyer, Linda Esker, Davide Falessi, Xavier Franch, Thomas Fritz, Nikolas Galanis, Marco Aurélio Gerosa, Ruediger Glott, Michael W. Godfrey, Alessandra Gorla, Georgios Gousios, Florian Groß, Randy Hackbarth, Abram Hindle, Reid Holmes, Lingxiao Jiang, Ron S. Kenett, Ekrem Kocaguneli, Oleksii Kononenko, Kostas Kontogiannis, Konstantin Kuznetsov, Lucas Layman, Christian Lindig, David Lo, Fabio Mancinelli, Serge Mankovskii, Shahar Maoz, Daniel Méndez Fernández, Andrew Meneely, Audris Mockus, Murtuza Mukadam, Brendan Murphy, Emerson Murphy-Hill, John Mylopoulos, Anil R. Nair, Maleknaz Nayebi, Hoan Nguyen, Tien Nguyen, Gustavo Ansaldi Oliva, John Palframan, Hridesh Rajan, Peter C. Rigby, Guenther Ruhe, Michele Shaw, David Shepherd, Forrest Shull, Will Snipes, Diomidis Spinellis, Eleni Stroulia, Angelo Susi, Lin Tan, Ilaria Tavecchia, Ayse Tosun Misirli, Mohsen Vakilian, Stefan Wagner, Shaowei Wang, David Weiss, Laurie Williams, Hamzeh Zawawy, and Andreas Zeller

The Art and Science of Analyzing Software Data

Edited by

Christian Bird, Tim Menzies, Thomas Zimmermann

Perspectives on Data Science for Software Engineering

Edited by Tim Menzies, Laurie Williams, Thomas Zimmermann

Contributors: Bram Adams, A. Bacchelli, T. Barik, E.T. Barr, O. Baysal, A. Bener, G. Bergersen, C. Bird, D. Budgen, B. Caglayan, T. Carnahan, J. Czerwonka, P. Devanbu, M. Di Penta, S. Diehl, T. Dyba, T. Fritz, M.W. Godfrey, G. Gousios, P. Guo, K. Herzig, A. Hindle, R. Holmes, Zhitao Hou, J. Huang, Andrew J. Ko, N. Juristo, S. Just, M. Kim, E. Kocaguneli, K. Kuutti, Qingwei Lin, Jian-Guang Lou, N. Medvidovic, A. Meneely, T. Menzies, L.L. Minku, A. Mockus, J. Munch, G.C. Murphy, B. Murphy, E. Murphy-Hill, M. Nagappan, M. Nayebi, M. Oivo, A. Orso, T. Ostrand, F. Peters, D. Posnett, L. Prechelt, Venkatesh-Prasad Ranganath, B. Ray, R. Robbes, P. Rotella, G. Ruhe, P. Runeson, B. Russo, M. Shepperd, E. Shihab, D.I.K. Sjøberg, D. Spinellis, M.-A. Storey, C. Theisen, A. Tosun, B. Turhan, H. Valdivia-Garcia, S. Vegas, S. Wagner, E. Weyuker, J. Whitehead, L. Williams, Tao Xie, A. Zeller, Dongmei Zhang, Hongyu Zhang, Haidong Zhang, T. Zimmermann

http://tiny.cc/superdog

tom’s data science research

2010-2012:
Information Needs
for **Analytics Tools**

FOSER 2010
ICSE 2012

2012-2014:
Questions that
Software Engineers have
for Data Scientists

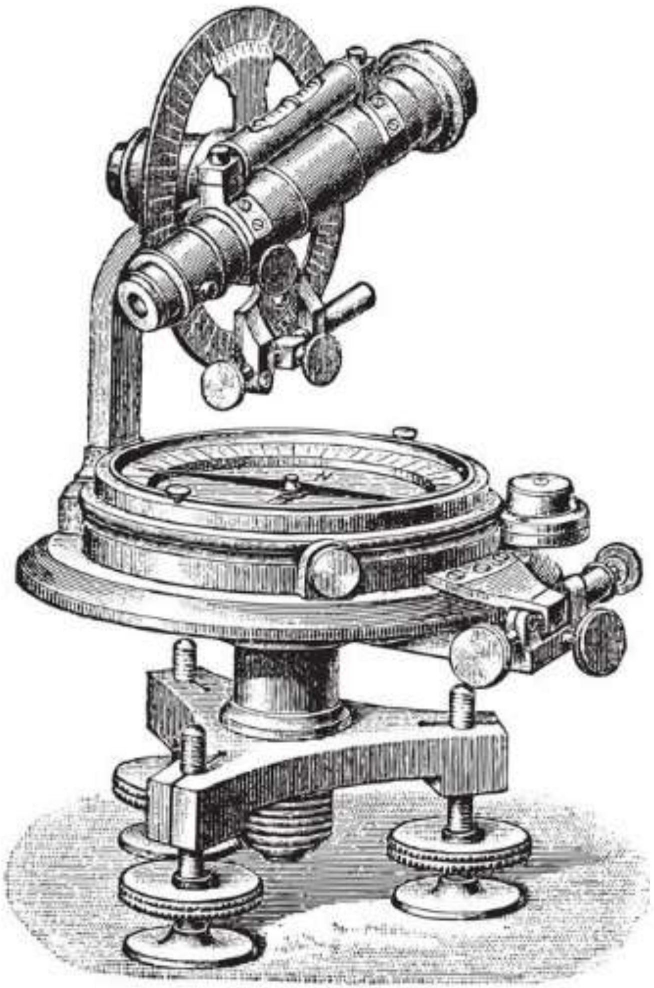
ICSE 2014

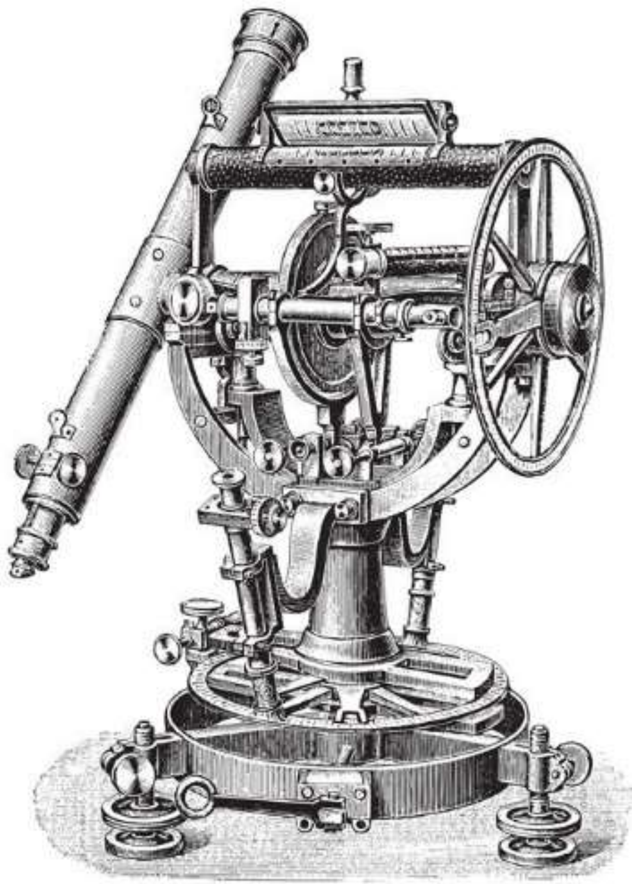
2014-now
The Emerging Role of
Data Scientists

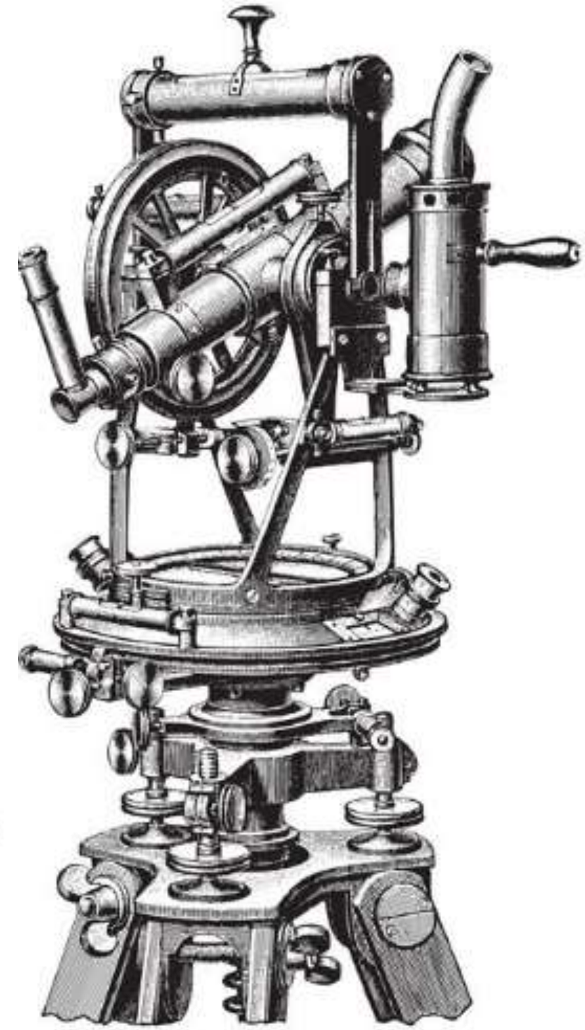
ICSE 2016

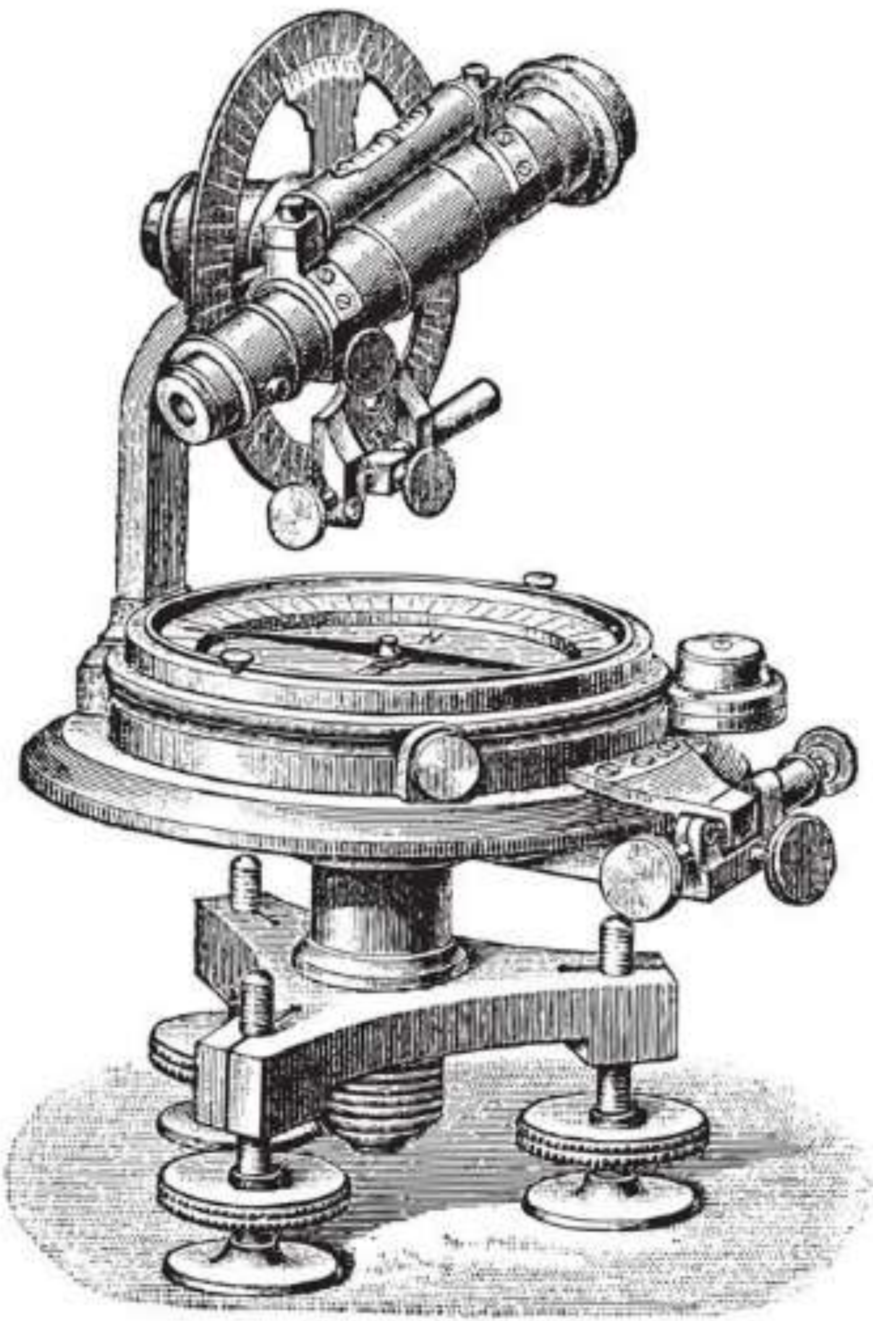

MARINERS
Microsoft

the lenses of data science









The Lens of

PEOPLE

ARTWORK: TAMAR COHEN, ANDREW J BUBOLTZ, 2011, SILK SCREEN ON A PAGE FROM A HIGH SCHOOL YEARBOOK, 8.5" X 12"

DATA

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

WHAT TO READ NEXT

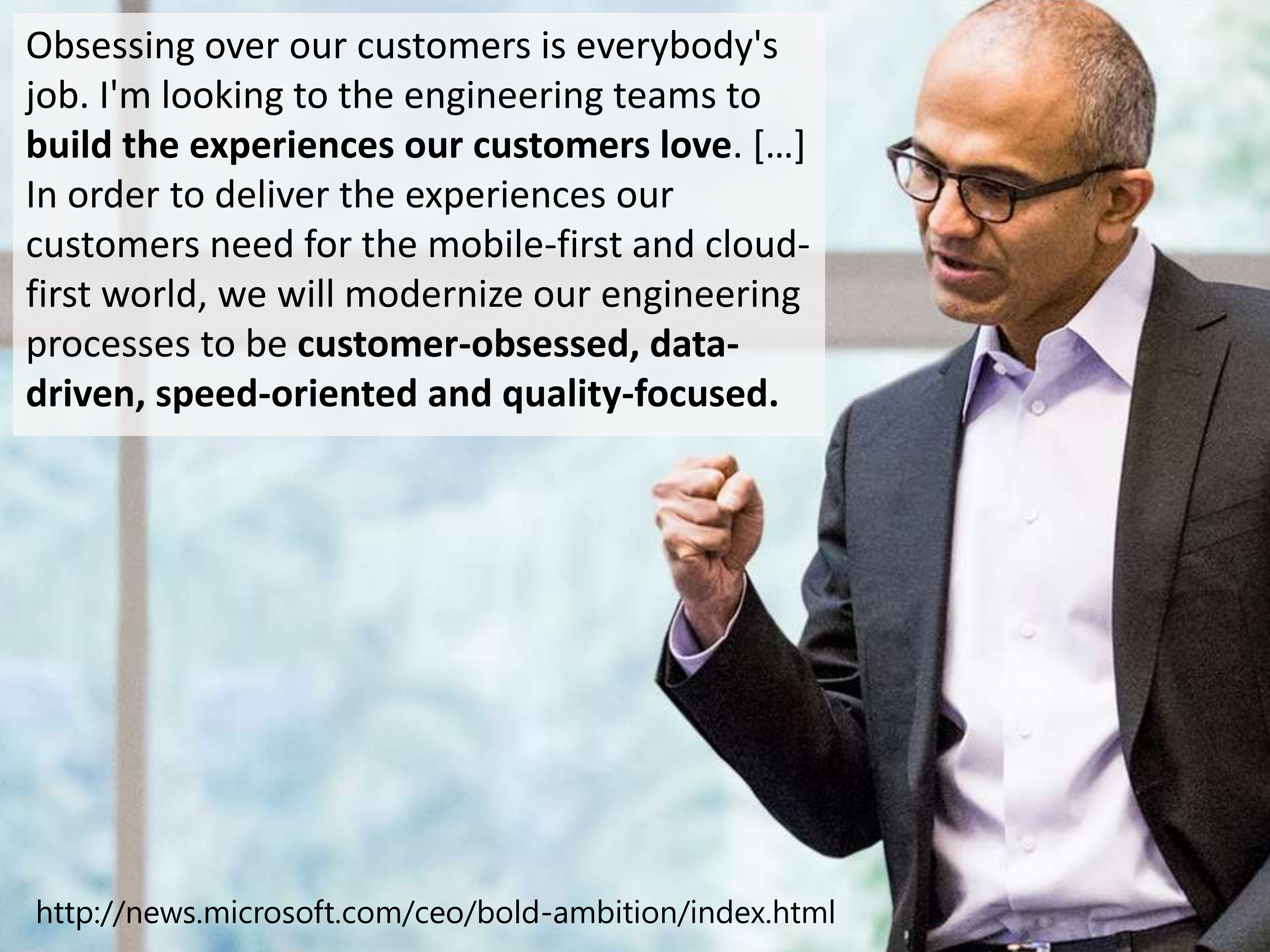
Big Data: The Management Revolution

Making Advanced Analytics Work for You

Google Flu Trends' Failure Shows Good Data > Big Data

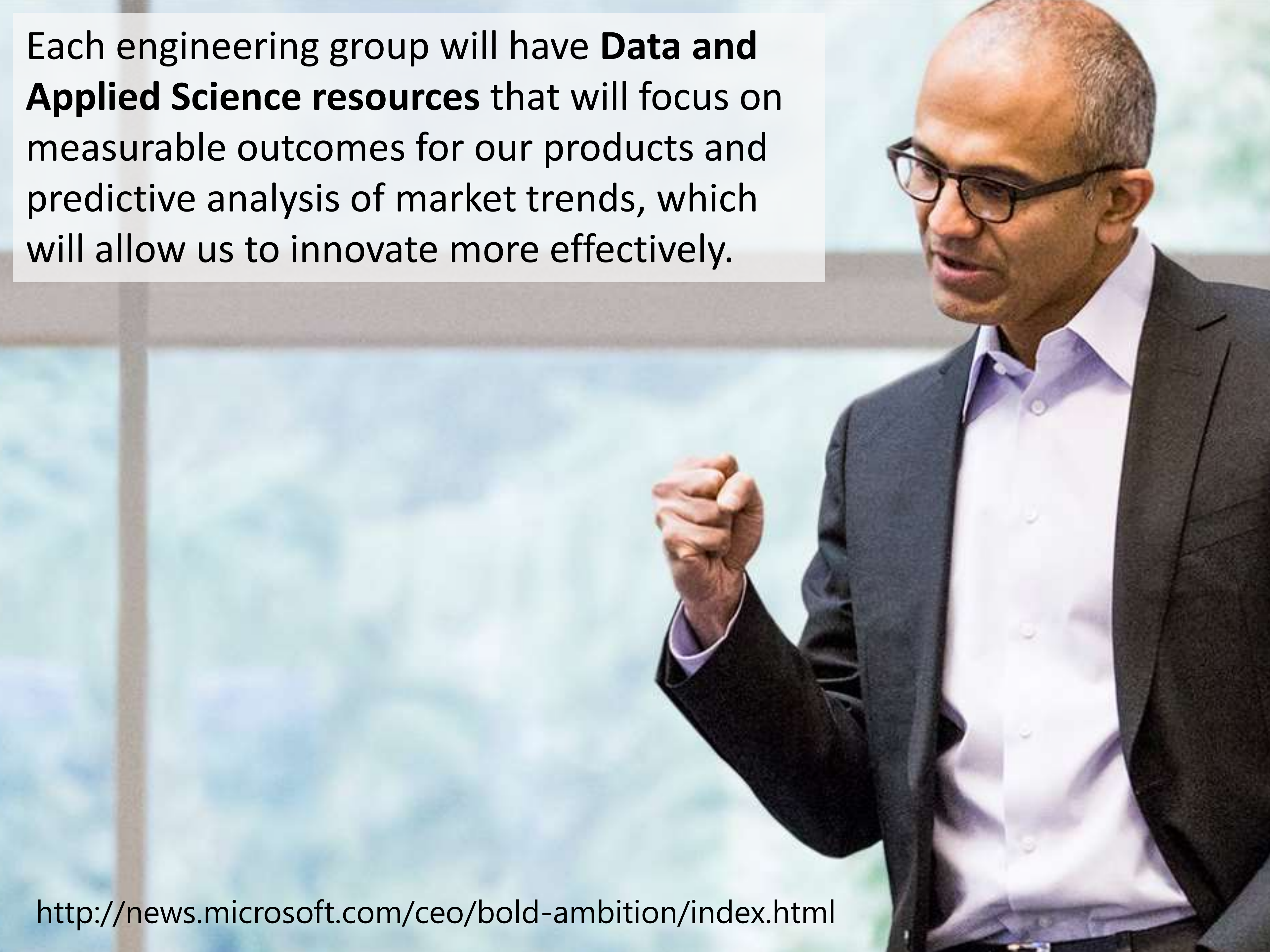
VIEW MORE FROM THE

October 2012 Issue



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

Robert DeLine

Andrew Begel

Miryung Kim

Miryung Kim, Thomas Zimmermann, Robert DeLine, Andrew Begel:
The Emerging Role of Data Scientists on Software Development Teams. ICSE 2016.

Methodology

- Interviews with 16 participants
  - 5 women and 11 men from eight different organizations at Microsoft
- Snowball sampling
  - data-driven engineering meet-ups and technical community meetings
  - word of mouth
- Coding with Atlas.TI
- Clustering of participants

Background of Data Scientists

Most CS, many interdisciplinary backgrounds

Many have higher education degrees

Strong passion for data

I love data, looking and making sense of the data. [P2]

I've always been a data kind of guy. I love playing with data. I'm very focused on how you can organize and make sense of data and being able to find patterns. I love patterns. [P14]

"Machine learning hackers". Need to know stats

My people have to know statistics. They need to be able to answer sample size questions, design experiment questions, know standard deviations, p-value, confidence intervals, etc.

Background of Data Scientists

PhD training contributes to working style

It has never been, in my four years, that somebody came and said, “Can you answer this question?” I mostly sit around thinking, “How can I be helpful?” Probably that part of your PhD is you are figuring out what is the most important questions. [P13]

I have a PhD in experimental physics, so pretty much, I am used to designing experiments. [P6]

Doing data science is kind of like doing research. It looks like a good problem and looks like a good idea. You think you may have an approach, but then maybe you end up with a dead end. [P5]

Working Styles of Data Scientists

Insight Provider

Specialists

Platform Builder

Polymath

Team Leader

Insight Providers

Insight Providers

Play an interstitial role between managers and engineers within a product group

Generate insights and to support and guide their managers in decision making

Analyze product and customer data collected by the teams’ engineers

Strong background in statistics

Communication and coordination skills are key

Insight Providers

P2 worked on a product line to inform managers needed to know whether an upgrade was of sufficient quality to push to all products in the family.

> *It should be as good as before. It should not deteriorate any performance, customer user experience that they have. Basically people shouldn't know that we've even changed [it].*

Insight Providers

Getting data from engineers

I basically tried to eliminate from the vocabulary the notion of "You can just throw the data over the wall ... She'll figure it out." There's no such thing.

I'm like, "Why did you collect this data? Why did you measure it like that? Why did you measure this many samples, not this many? Where did this all come from?"

Modelling Specialists

Modelling Specialists

Act as expert consultants

Build predictive models that can be instantiated as new software features and support other team's data-driven decision making

Strong background in machine learning

Other forms of expertise such as survey design or statistics would fit as well

Modelling Specialists

P7 is an expert in time series analysis and works with a team on automatically detecting anomalies in their telemetry data.

> *The [Program Managers] and the Dev Ops from that team... through what they daily observe, come up with a new set of time series data that they think has the most value and then they will point us to that, and we will try to come up with an algorithm or with a methodology to find the anomalies for that set of time series.*

Platform Builders

Platform Builders

Build data engineering platforms
that are reusable in many contexts

Strong background in big data systems

Make trade-offs between engineering and scientific concerns

Platform Builders

P4 worked on platform to collect crash data.

> *You come up with something called a bucket feed. It is a name of a function most likely responsible for the crash in the small bucket.*
> *We found in the source code who touch last time this function. He gets the bug.*
> *And we filed [large] numbers a year with [a high] percent fix rate.*

Polymaths

Polymaths

Data scientists who “do it all”:

- Forming a business goal
- Instrumenting a system to collect data
- Doing necessary analyses or experiments
- Communicating the results to managers

Polymaths

P13 works on a product that serves ads and explores her own ideas for new data models.

> So I am the only scientist on this team. I'm the only scientist on sort of sibling teams and everybody else around me are like just straight-up engineers.
>
> For months at a time I'll wear a dev hat and I actually really enjoy that, too. ... I spend maybe three months doing some analysis and maybe three months doing some coding that is to integrate whatever I did into the product. … I do really, really like my role. I love the flexibility that I can go from being developer to being an analyst and kind of go back and forth.

Team Leaders

Team Leaders

Senior data scientists who typically run their own data science teams

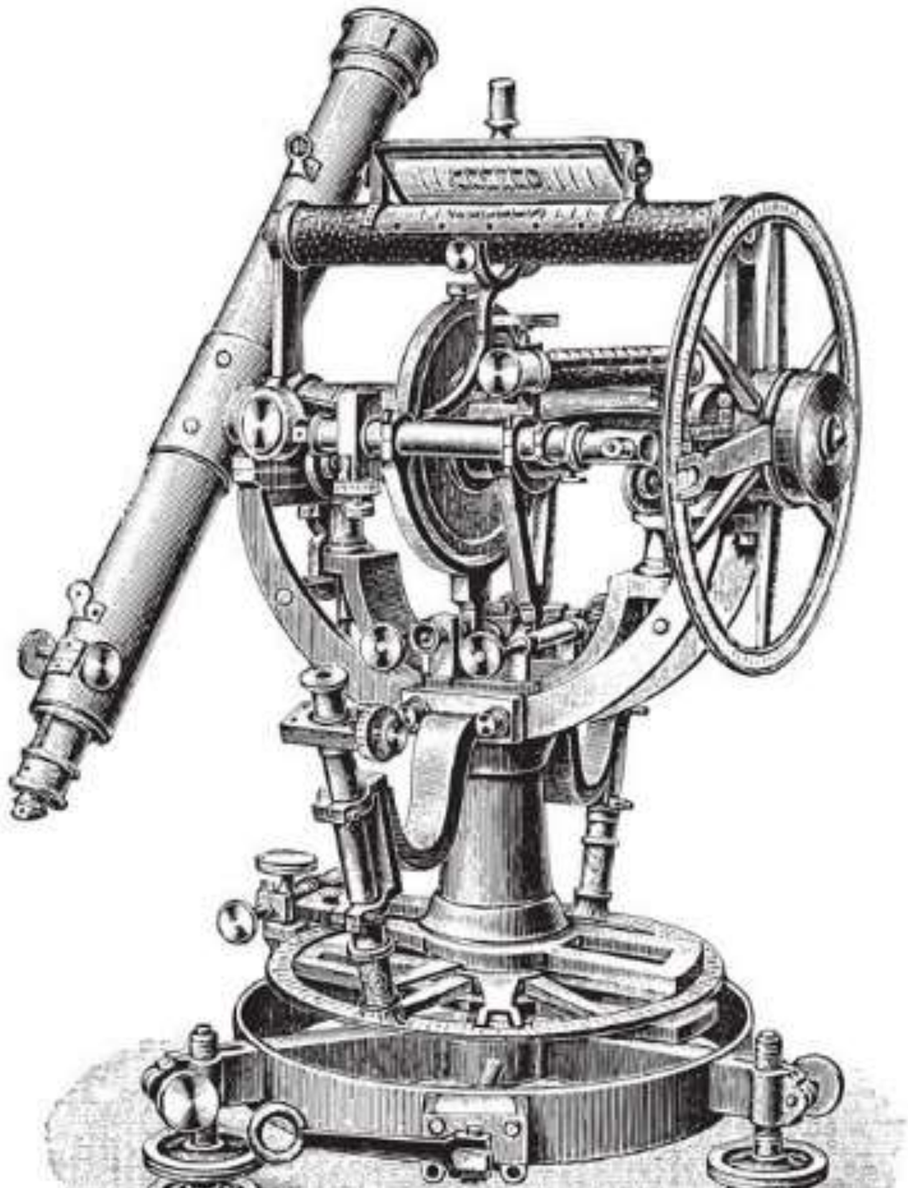
Act as data science “evangelists”, pushing for the adoption of data-driven decision making

Work with senior company leaders to inform broad business decisions

Team Leaders

P10 and his team of data scientists estimated the number of bugs that would remain open when a product was scheduled to ship.

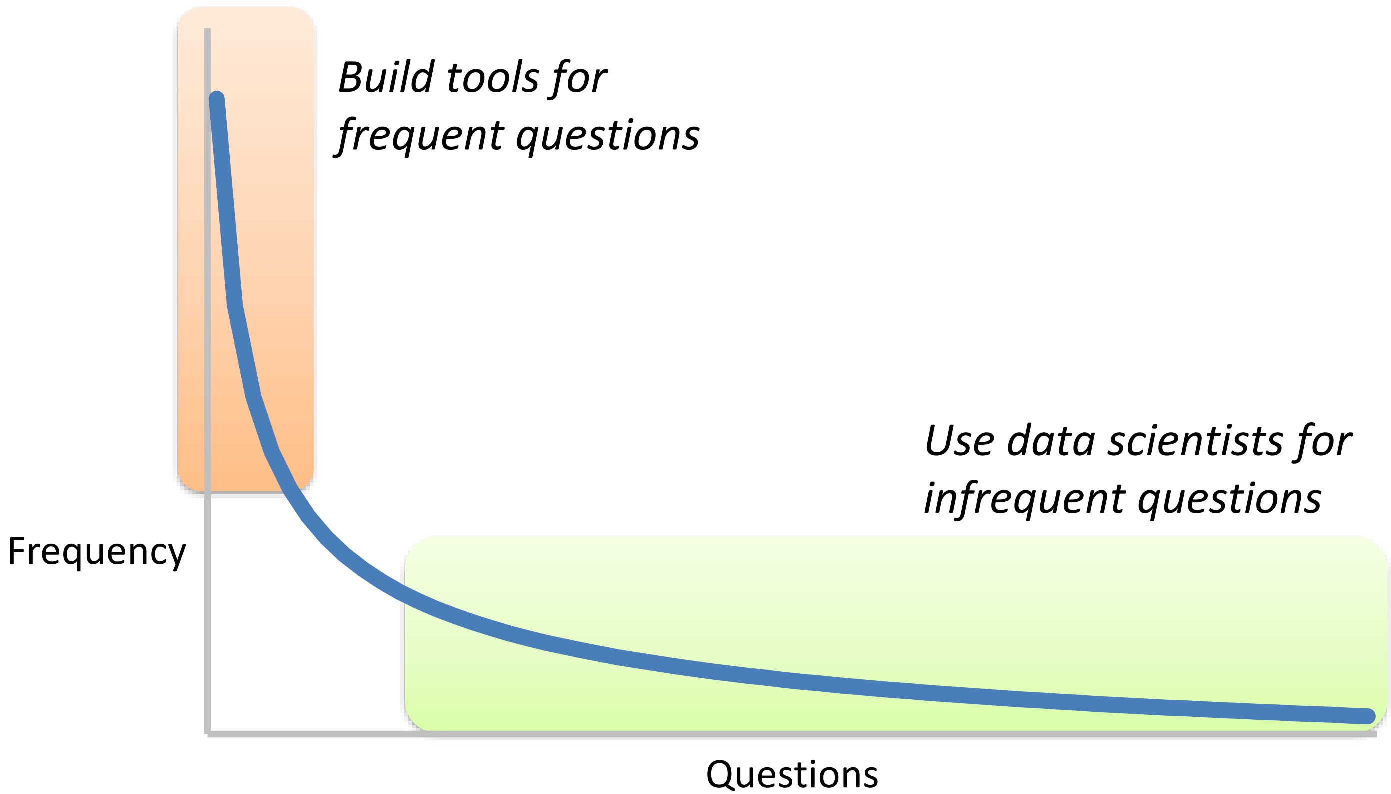
> *When the leadership saw this gap [between the estimated bug count and the goal], the allocation of developers towards new features versus stabilization shifted away from features toward stabilization to get this number back.*
>
> *Sometimes people who are real good with numbers are not as good with words (laughs), and so having an intermediary to sort of handle the human interfaces between the data sources and the data scientists, I think, is a way to have a stronger influence. [Acting] an intermediary so that the scientists can kind of stay focused on the data.*



The Lens of

QUESTIONS

The Long Tail of Questions



Andrew Begel

Andrew Begel, Thomas Zimmermann:
Analyze this! 145 questions for data scientists in software engineering. ICSE 2014

Meet Greg Wilson from Mozilla

It Will Never Work in Theory

Ten Questions for Researchers

Posted Aug 22, 2012 by Greg Wilson

I gave the opening talk at [MSR Vision 2020] in Kingston on Monday ([slides]), and in the wake of that, an experienced developers at Mozilla sent me a list of ten questions he'd really like empirical software engineering researchers to answer. They're interesting in their own right, but I think they also reveal a lot about what practitioners want from researchers in general; comments would be very welcome.

1. Vi vs. Emacs vs. graphical editors/IDEs: which makes me more productive?
2. Should language developers spend their time on tools, syntax, library, or something else (like speed)? What makes the most difference to their users?
3. Do unit tests save more time in debugging than they take to write/run/keep updated?

Let’s ask Microsoft engineers what they would like to know!

http://aka.ms/**145Questions**

1

SURVEY

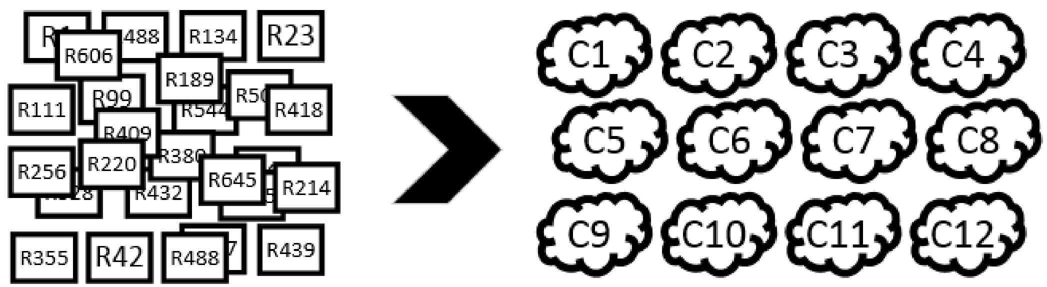
203 participants, 728 questions R1..R728

Suppose you could work with a team of data scientists and data analysts who specialize in studying how software is developed. Please list up to five questions you would like them to answer.

✶ CATEGORIES

12 categories C1..C12

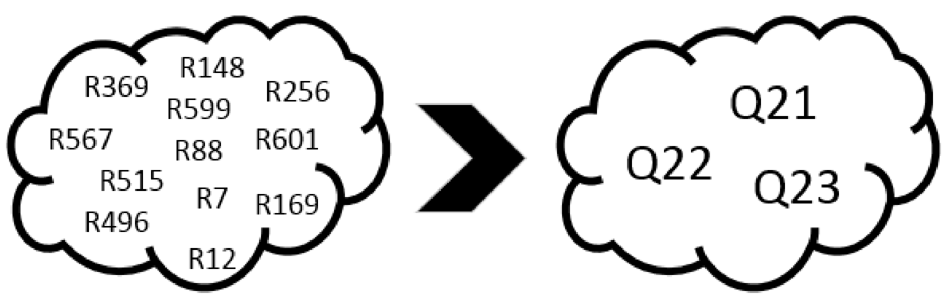
Use an open card sort to group questions into categories.

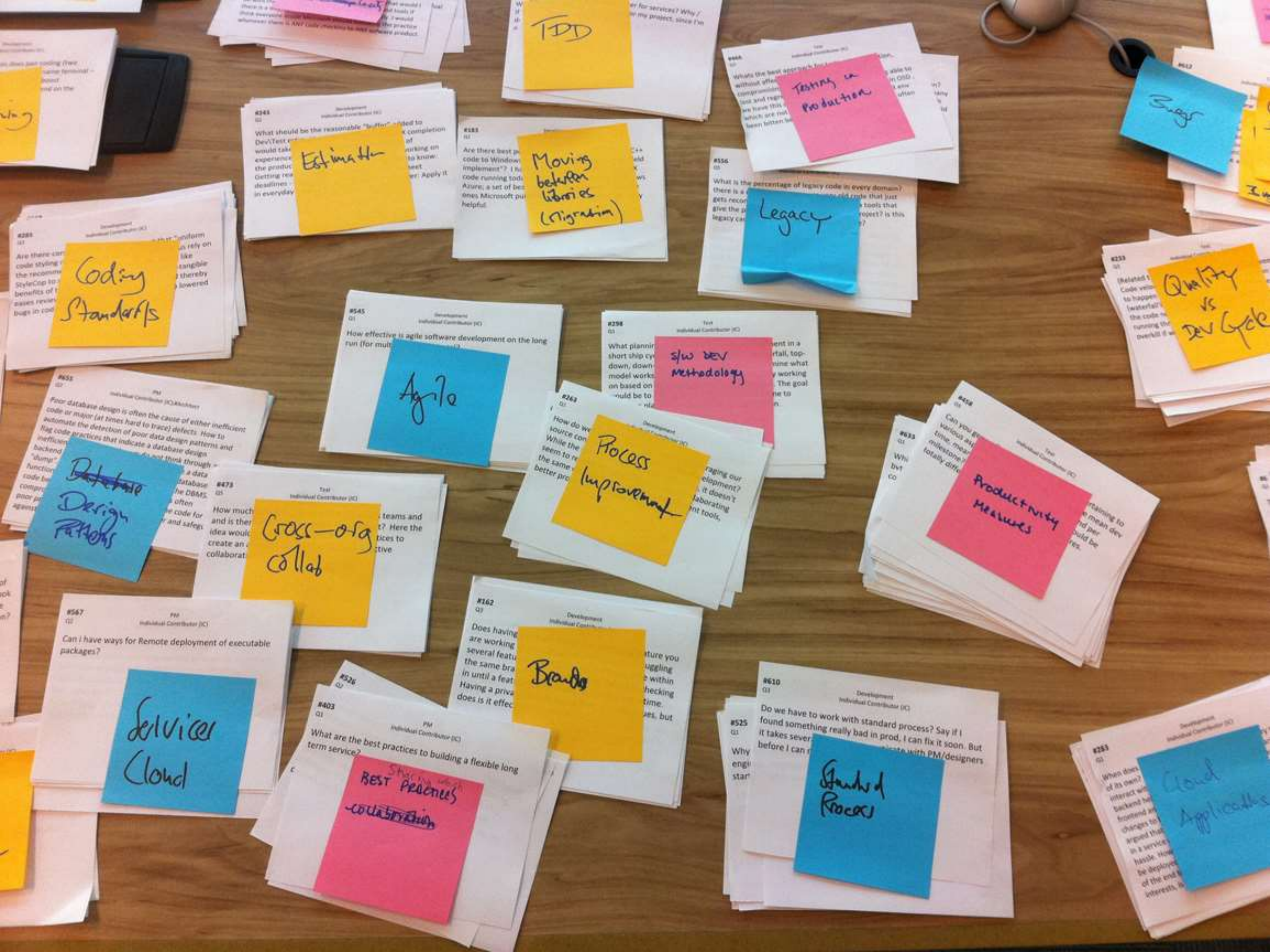


✶ DESCRIPTIVE QUESTIONS

145 questions Q1..Q145

Summarize each category with a set of descriptive questions.




TDD
Testing in Production
Bugs
Estimation
Moving between libraries (migration)
Legacy
Coding Standards
Quality vs Dev Cycle
Agile
S/W Dev Methodology
Process Improvement
Productivity Measures
~~Database~~ Design Patterns
Cross-org collab
Services Cloud
Standard Process
Best Practices
Cloud Applicable

raw questions (provided by the respondents)

"How does the quality of software change over time – does software age? I would use this to plan the replacement of components."

"How do security vulnerabilities correlate to age / complexity / code churn / etc. of a code base? Identify areas to focus on for in-depth security review or re-architecting."

"What will the cost of maintaining a body of code or particular solution be? Software is rarely a fire and forget proposition but usually has a fairly predictable lifecycle. We rarely examine the long term cost of projects and the burden we place on ourselves and SE as we move forward."

raw questions (provided by the respondents)

"How does the quality of software change over time – does software age? I would use this to plan the replacement of components."

"How do security vulnerabilities correlate to age / complexity / code churn / etc. of a code base? Identify areas to focus on for in-depth security review or re-architecting."

"What will the cost of maintaining a body of code or particular solution be? Software is rarely a fire and forget proposition but usually has a fairly predictable lifecycle. We rarely examine the long term cost of projects and the burden we place on ourselves and SE as we move forward."

descriptive question (which we distilled)

How does the age of code affect its quality, complexity, maintainability, and security?

2

SURVEY

607 participants, 16 765 ratings

Split questionnaire design, where each participant received a subset of the questions Q1..Q145 (on average 27.6) and was asked:

In your opinion, how important is it to have a software data analytics team answer this question?
[Essential | Worthwhile | Unimportant | Unwise | I don't understand]

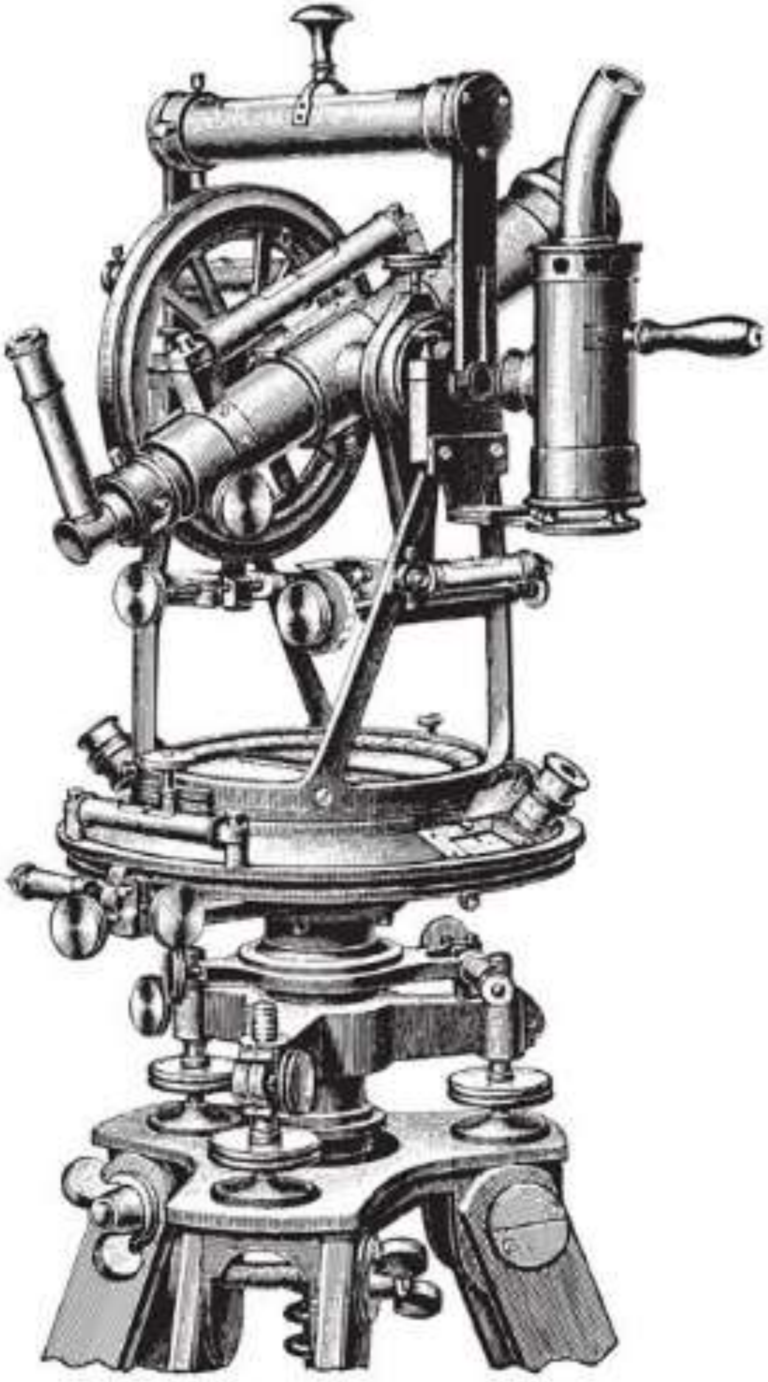
✶ TOP/BOTTOM RANKED QUESTIONS

✶ DIFFERENCES IN DEMOGRAPHICS

Discipline: Development, Testing, Program Management
Region: Asia, Europe, North America, Other
Number of Full-Time Employees
Current Role: Manager, Individual Contributor
Years as Manager
Has Management Experience: yes, no.
Years at Microsoft

Microsoft’s Top 10 Questions	Essential	Essential + Worthwhile
How do users typically use my application?	80.0%	99.2%
What parts of a software product are most used and/or loved by customers?	72.0%	98.5%
How effective are the quality gates we run at checkin?	62.4%	96.6%
How can we improve collaboration and sharing between teams?	54.5%	96.4%
What are the best key performance indicators (KPIs) for monitoring services?	53.2%	93.6%
What is the impact of a code change or requirements change to the project and its tests?	52.1%	94.0%
What is the impact of tools on productivity?	50.5%	97.2%
How do I avoid reinventing the wheel by sharing and/or searching for code?	50.0%	90.9%
What are the common patterns of execution in my application?	48.7%	96.6%
How well does test coverage correspond to actual code usage by our customers?	48.7%	92.0%

Microsoft's 10 Most Unwise Questions	Unwise
Which individual measures correlate with employee productivity (e.g. employee age, tenure, engineering skills, education, promotion velocity, IQ)?	25.5%
Which coding measures correlate with employee productivity (e.g. lines of code, time it takes to build software, particular tool set, pair programming, number of hours of coding per day, programming language)?	22.0%
What metrics can use used to compare employees?	21.3%
How can we measure the productivity of a Microsoft employee?	20.9%
Is the number of bugs a good measure of developer effectiveness?	17.2%
Can I generate 100% test coverage?	14.4%
Who should be in charge of creating and maintaining a consistent company-wide software process and tool chain?	12.3%
What are the benefits of a consistent, company-wide software process and tool chain?	10.4%
When are code comments worth the effort to write them?	9.6%
How much time and money does it cost to add customer input into your design?	8.3%



The Lens of

RELEVANCE


Papers
Research
Industry

Take your time to defining ground truth

*You have **communication going back and forth where you will find what you're actually looking for,** what is anomalous and what is not anomalous in the set of data that they looked at.*

Operationalization of models is important

They accepted [the model] and they understood all the results and they were very excited about it. Then, there's a ***phase that comes in where the actual model has to go into production.*** *… You really need to have somebody who is confident enough to take this from a dev side of things.*

Translate findings into business values

*In terms of convincing, if you **just present all these numbers like precision and recall factors**… that is important from the knowledge sharing model transfer perspective. But if you are out there to sell your model or ideas, this **will not work because the people who will be in the decision-making seat will not be the ones doing the model transfer.** So, for those people, what we did is cost benefit analysis where we showed how our model was adding the new revenue on top of what they already had.*

Choose the right questions for the right team

*(a) Is it a **priority** for the organization*

*(b) is it **actionable**, if I get an answer to this, is this something someone can do something with? and,*

*(c) are you as the feature team — if you're coming to me or if I'm going to you, telling you this is a good opportunity — are you **committing resources** to deliver a change?*

If those things are not true, then it's not worth us talking anymore.

Work closely with consumers from day one

You begin to find out, you begin to ask questions, you being to see things. And so ***you need that interaction with the people that own the code,*** *if you will, or the feature, to be able to learn together as you go and refine your questions and refine your answers to get to the ultimate insights that you need.*

Explain the findings in simple terms

A super smart data scientist, their understanding and presentation of their findings is usually way over the head of the managers…so my guidance to [data scientists], is ***dumb everything down to seventh-grade level****, right? And whether you're writing or you're presenting charts, you know, keep it simple.*

Nachi Nagappan

Jeff Carver

David Lo

Oscar Dieste

Nicolas Kraft

David Lo, Nachiappan Nagappan, Thomas Zimmermann: How practitioners perceive the relevance of software engineering research. ESEC/SIGSOFT FSE 2015: 415-425

Jeff Carver, Oscar Dieste, Nicholas Kraft, David Lo, Thomas Zimmermann:
How practitioners perceive the relevance of ESEM research. ESEM 2016

Feedback-Driven Conferences

Survey a representative group of practitioners for feedback on papers

Feedback-Driven Conferences

Organizers

Assess/improve industrial relevance

Publicity for the conference

Authors

Additional feedback on research

More visibility

Practitioners

Overview of latest research

Proof-Of-Concept

Summarize 571 Papers from five years of ICSE, ESEC/FSE and FSE conferences.

Cross-project Defect Prediction

A Large Scale Experiment on Data vs. Domain vs. Process

Thomas Zimmermann, Microsoft Research

Nachiappan Nagappan, Microsoft Research

Harald Gall, University of Zurich

Emanuel Giger, University of Zurich

Brendan Murphy, Microsoft Research

Empirical study of using software defect data from one project to predict defects in another project.

Proof-Of-Concept

In your opinion, how important are the following pieces of research? Please respond to as many as possible. (at least 1 response is required)*

	Essential	Worthwhile	Unimportant	Unwise	I don't understand
Empirical study of using software defect data from one project to predict defects in another project.					
Empirical study on whether the bug fixes recorded in these historical datasets is a fair representation of the full population of bug fixes.					

(40 randomly selected summaries)

Proof-Of-Concept

On the previous page, you selected the following research idea as "Unwise":

"Technique to identify bugs that contain a bug from a bug report."

To help us better understand your response, could you please explain why.

Response Statistics

3,000 randomly selected Microsoft practitioners working in technical roles

512 responded (17% response rate)

- developers (291), testers (87), and PMs (102)

17,913 ratings, 16-47 ratings per paper

173 responses why papers are “unwise”

Data Analysis

In your opinion, how important are the following pieces of research? Please respond to as many as possible. (at least 1 response is required)*

Essential	Worthwhile	Unimportant	Unwise	I don't understand

E-Score:

Proportion of ratings that are “Essential”

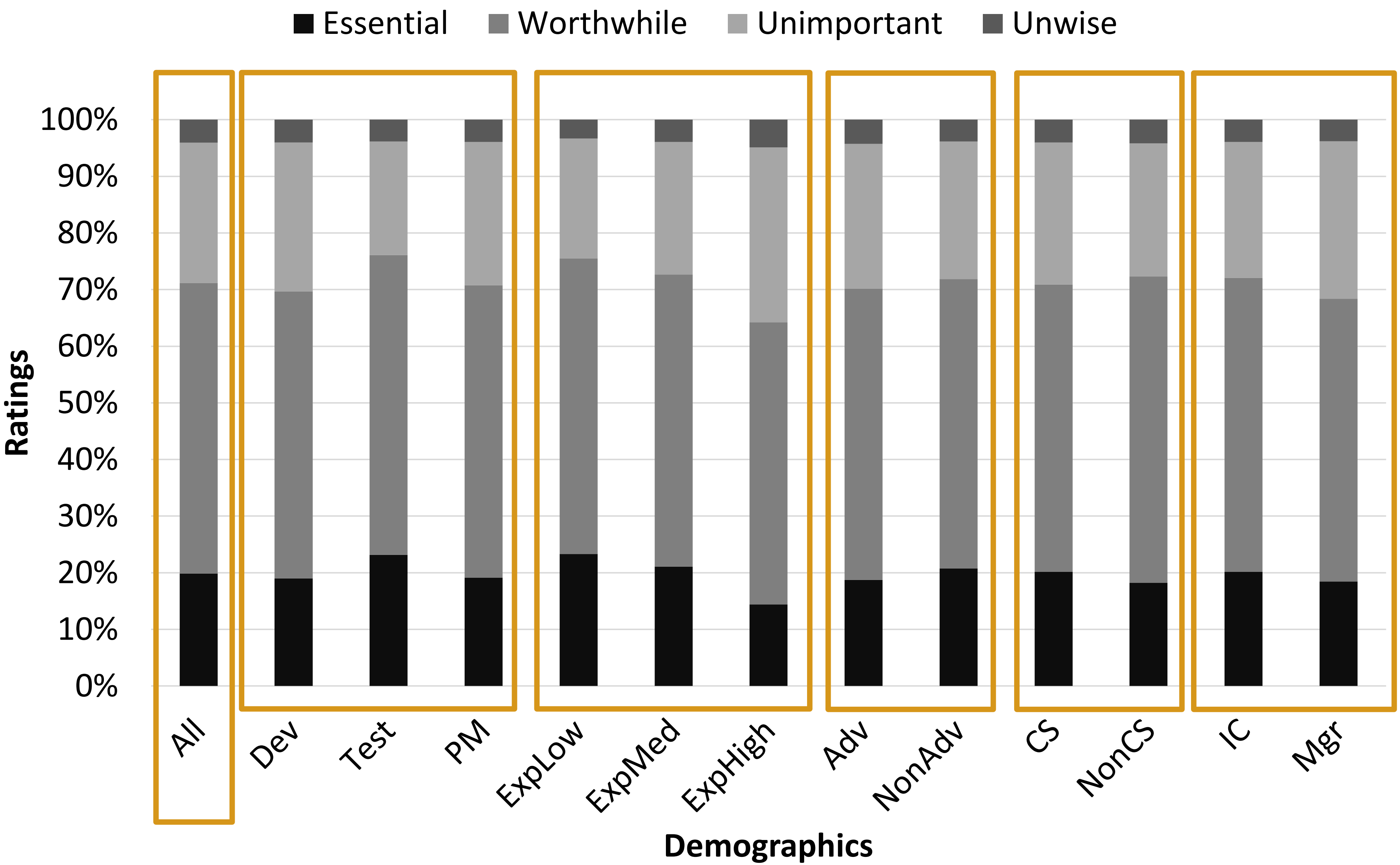
EW-Score:

Proportion of ratings that are “Essential” or “Worthwhile”

U-Score:

Proportion of ratings that are “Unwise”

Practitioner Perception



Highly Rated Research (1)

An approach to help developers identify and resolve conflicts early during collaborative software development, before those conflicts become severe and before relevant changes fade away in the developers' memories.

Technique that clusters call stack traces to help performance analysts effectively discover highly impactful performance bugs (e.g., bugs impacting many users with long response delay).

Symbolic analysis algorithm for buffer overflow detection that scale to millions of lines of code (MLOC) and can effectively handle loops and complex program structures.

Highly Rated Research (2)

Automatic generation of efficient multithreaded random tests that effectively trigger concurrency bugs.

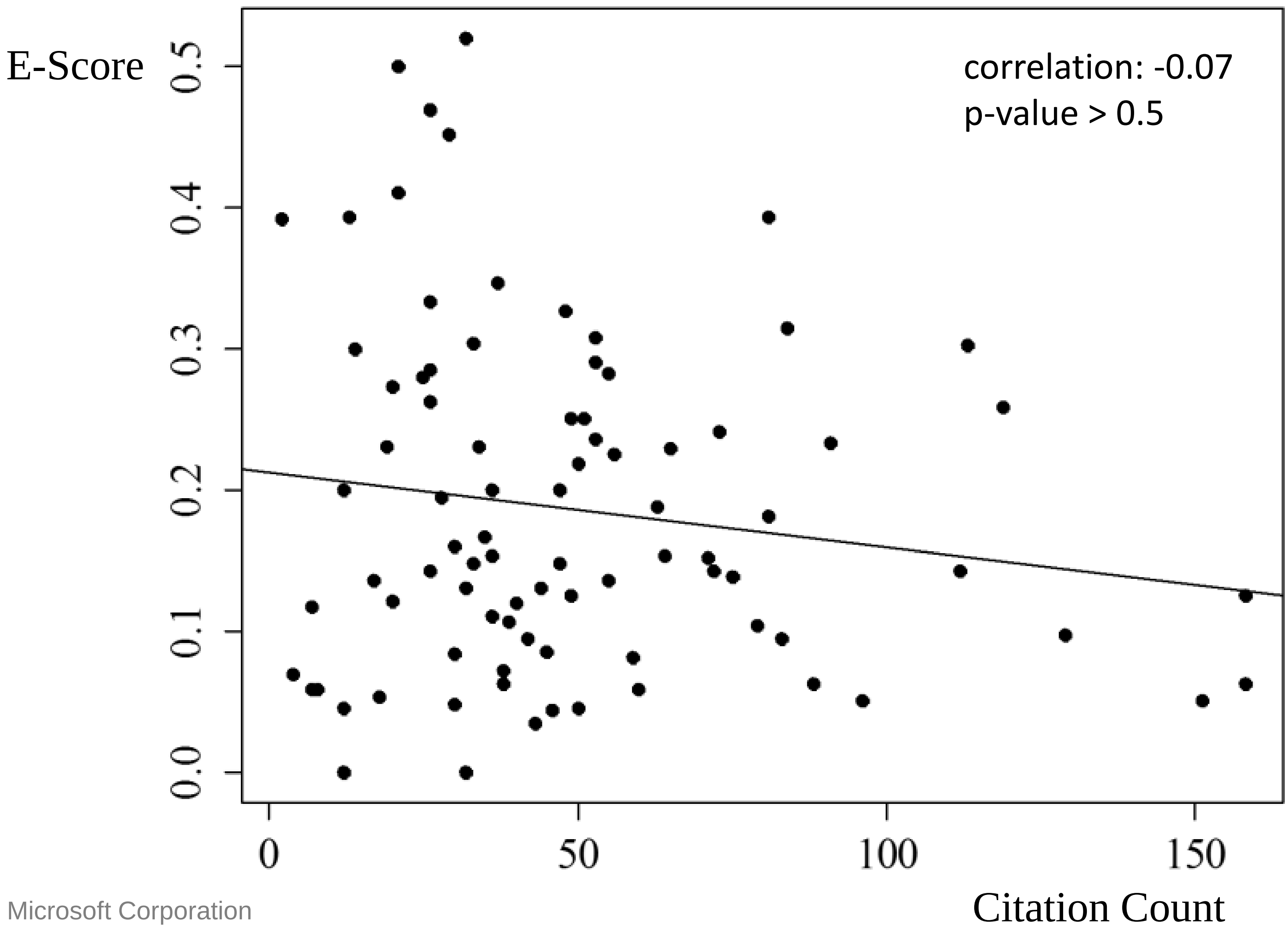
Debugging tool that uses objects as key abstractions to support debugging operations. Instead of setting breakpoints that refer to source code, one sets breakpoints with reference to a particular object.

Technique to make runtime reconfiguration of distributed systems in response to changing environments and evolving requirements safe and being done in a low-disruptive way through the concept of version consistency of distributed transactions.

Barriers to Relevance

- A tool is not needed
- An empirical study is not actionable
- Generalizability issue
- Cost outweighs benefit
- Questionable assumptions
- Disbelief in a particular technology/methodology
- Another solution seems better or another problem more important
- Proposed solution has side effects

E-Score vs. Citation Count



YOU HAVEN'T SEEN ANYTHING
UNTIL YOU'VE SEEN
EVERYTHING*

Call to Action: Researchers

Data scientists are *now* in software teams. They need your help!

Better techniques to analyze data.

New tools to automate the collection, analysis, and validation of data.

Translate research findings so that they can be easily consumed by industry.

Learn success strategies from data scientists.

Call to Action: Educators

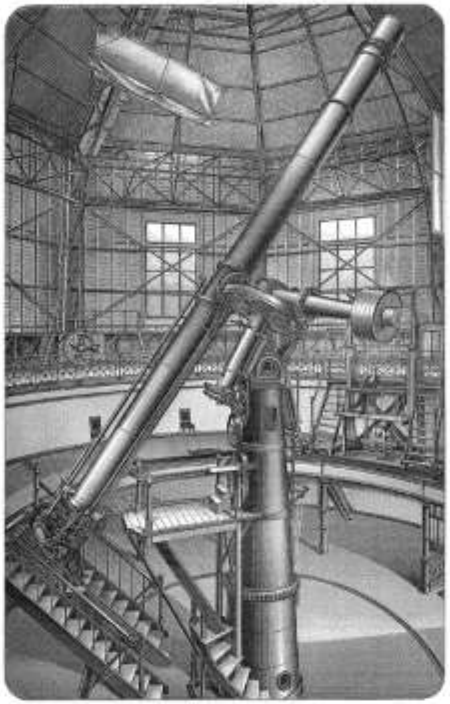
Industry needs more data scientists. :-)

Data science is not always a distinct role on the team; it is a skillset that often blends with other skills such as software development.

Data science requires many different skills.

Communication skills are very important.

Data scientists very similar to researchers.



The Lens of

PEOPLE
QUESTIONS
RELEVANCE

DATA

Working Styles of Data Scientists

Insight Provider

Specialists

Platform Builder

Polymath

Team Leader

Microsoft's Top 10 Questions	Essential	Essential + Worthwhile
How do users typically use my application?	80.0%	99.2%
What parts of a software product are most used and/or loved by customers?	72.0%	98.5%
How effective are the quality gates we run at checkin?	62.4%	96.6%
How can we improve collaboration and sharing between teams?	54.5%	96.4%
What are the best key performance indicators (KPIs) for monitoring services?	53.2%	93.6%
What is the impact of a code change or requirements change to the project and its tests?	52.1%	94.0%
What is the impact of tools on productivity?	50.5%	97.2%
How do I avoid reinventing the wheel by sharing and/or searching for code?	50.0%	90.9%
What are the common patterns of execution in my application?	48.7%	96.6%
How well does test coverage correspond to actual code usage by our customers?	48.7%	92.0%

Feedback-Driven Conferences

Organizers

Assess/improve industrial relevance

Publicity for the conference

Authors

Additional feedback on research

More visibility

Practitioners

Overview of latest research

Thank you!