GROWTH LESSONS FROM THE REFACTORING COMMUNITY

CSCI 7000-025
WEEK 1_L2

Danny Dig

University of Colorado
Boulder

YOUR expectations from CSCI 7000-025

A. How to communicate technical material to outsiders
B. How do we conduct industry-relevant research
C. Understand limitations and opportunities of IoT
D. Learn applications of IoT for human users
E. Whole picture vantage of IoT, how to process IoT data
F. Overview of IoT applications in healthcare, and the rest of the field
G. Common challenges across different applications and how we approach them with reuse

Boulder

Course Objectives

IoT research & practice

- **state of the art, state of the practice**

Develop critical thinking abilities

Practice giving scientific presentations and teaching others

Engage in active learning activities in class, such as paper discussions

Practice a research or novel-industrial project through all its stages

Have fun learning

Course Syllabus

Research-based course

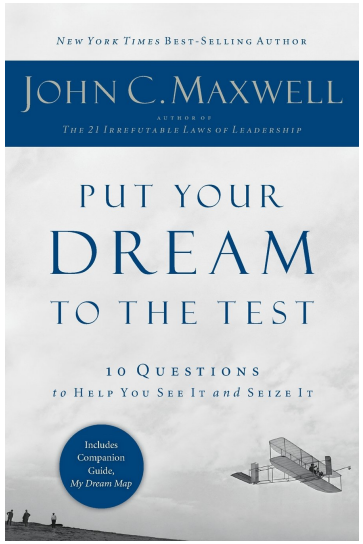
Complete a research or industrial-novel project of your choice (teams of 2-3 students) [50%]

Read papers, write Critiques. [20%]

Class Discussion [10%]

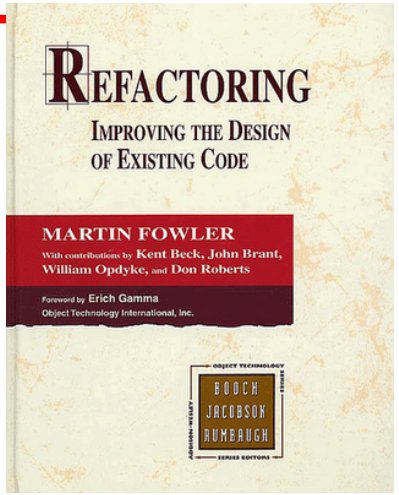
Papers Presentation [20%]

Put Your Dream to the Test

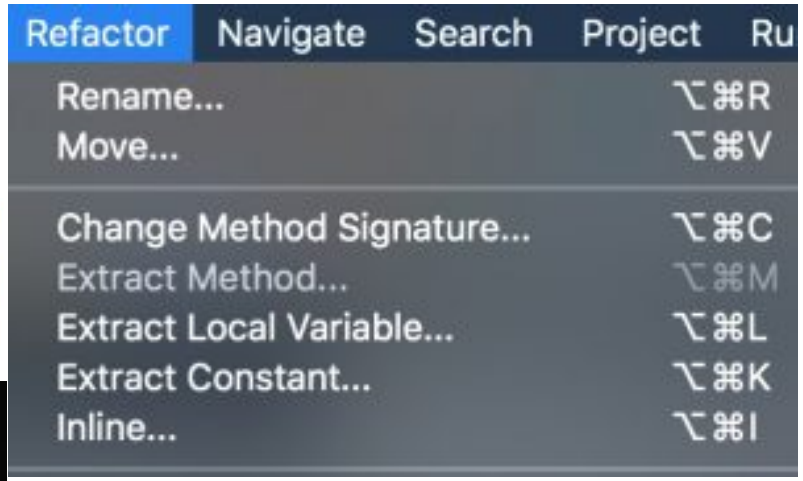


What is Refactoring?

"A change made to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behaviour" – M. Fowler [1999]



Top-level menu in all modern IDEs



In 2000, I created the first open-source refactoring tool

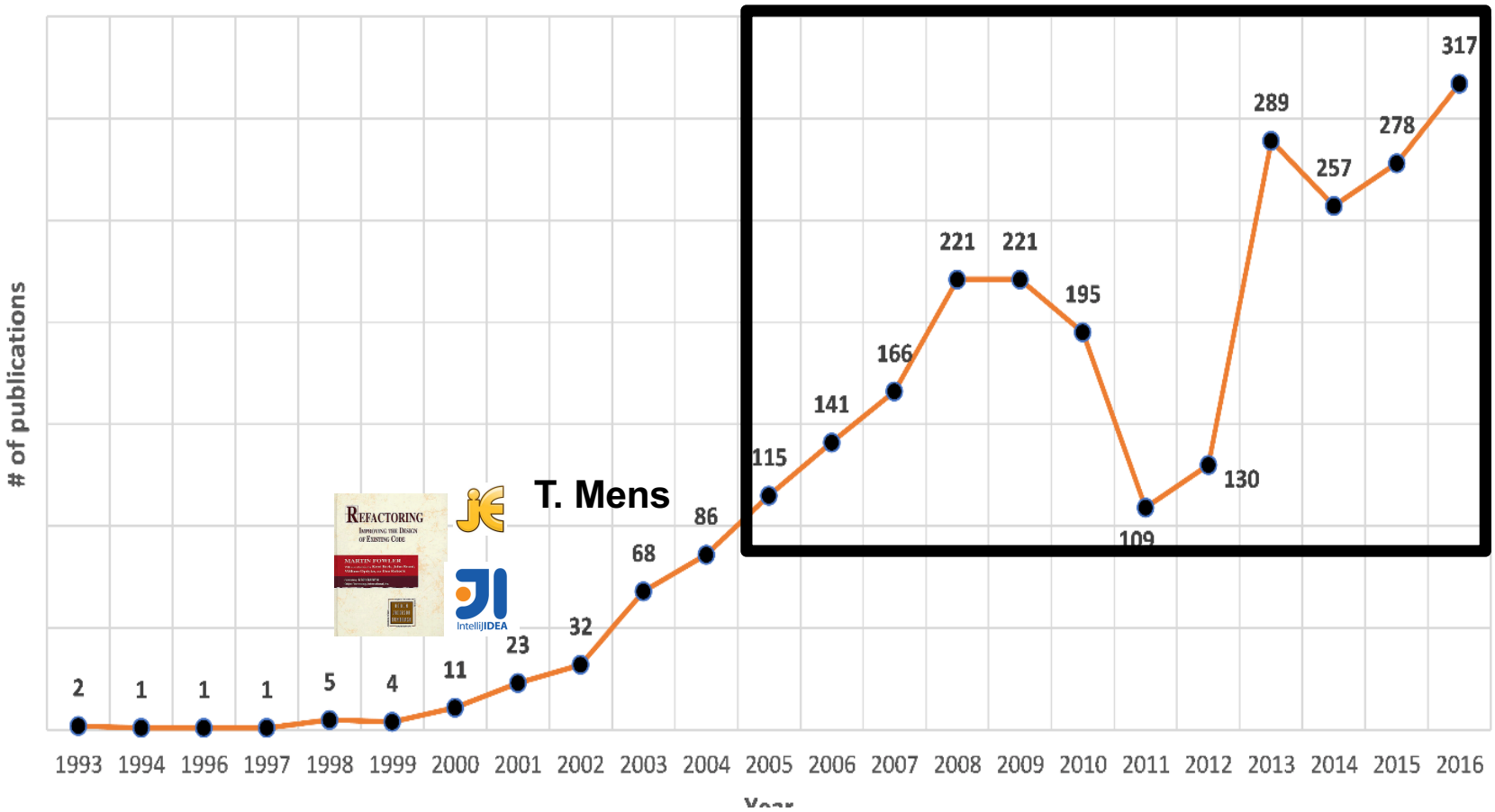
In 2004, I joined the team

To go fast, go alone. To go far, go with others.

A Decade of Refactoring Research

2,880 refactoring papers since 1990

2,442 papers between 2005-2016



Corpus of Papers

Work done by Marouane Kessentini and his team at Michigan

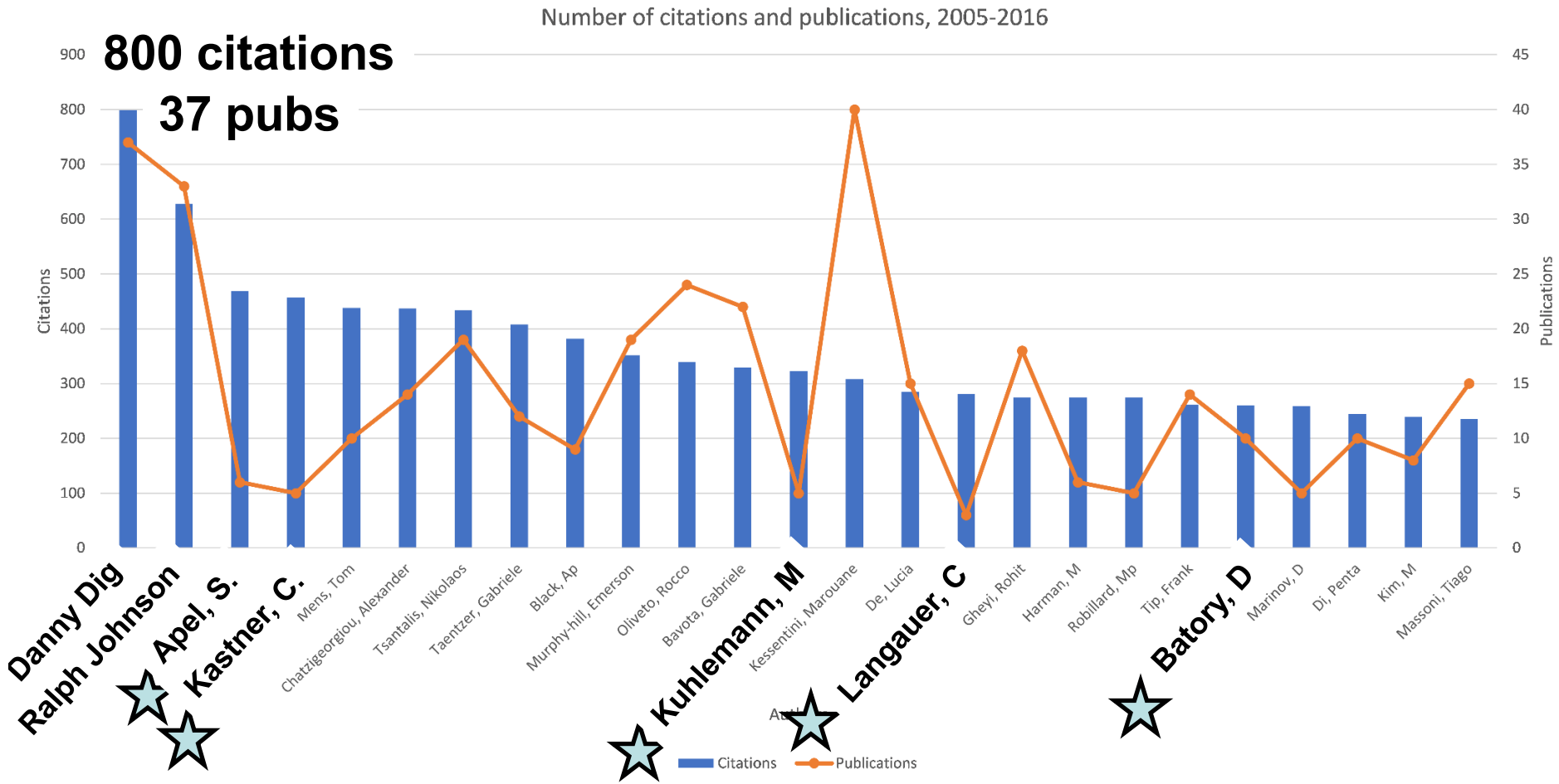
Scopus and Web of Science
- **"Refactoring" in title, abstract, and keywords**
- **yielded 3277 papers**

Refactoring definition:
- **transformation with behavior preservation**

Manual validation of ALL papers:
- **each paper analyzed title, abstract (and sometimes content)**
- **4 grad students who took a graduate class on Softw QA**
- **Kessentini (faculty) looked at the contentious papers**

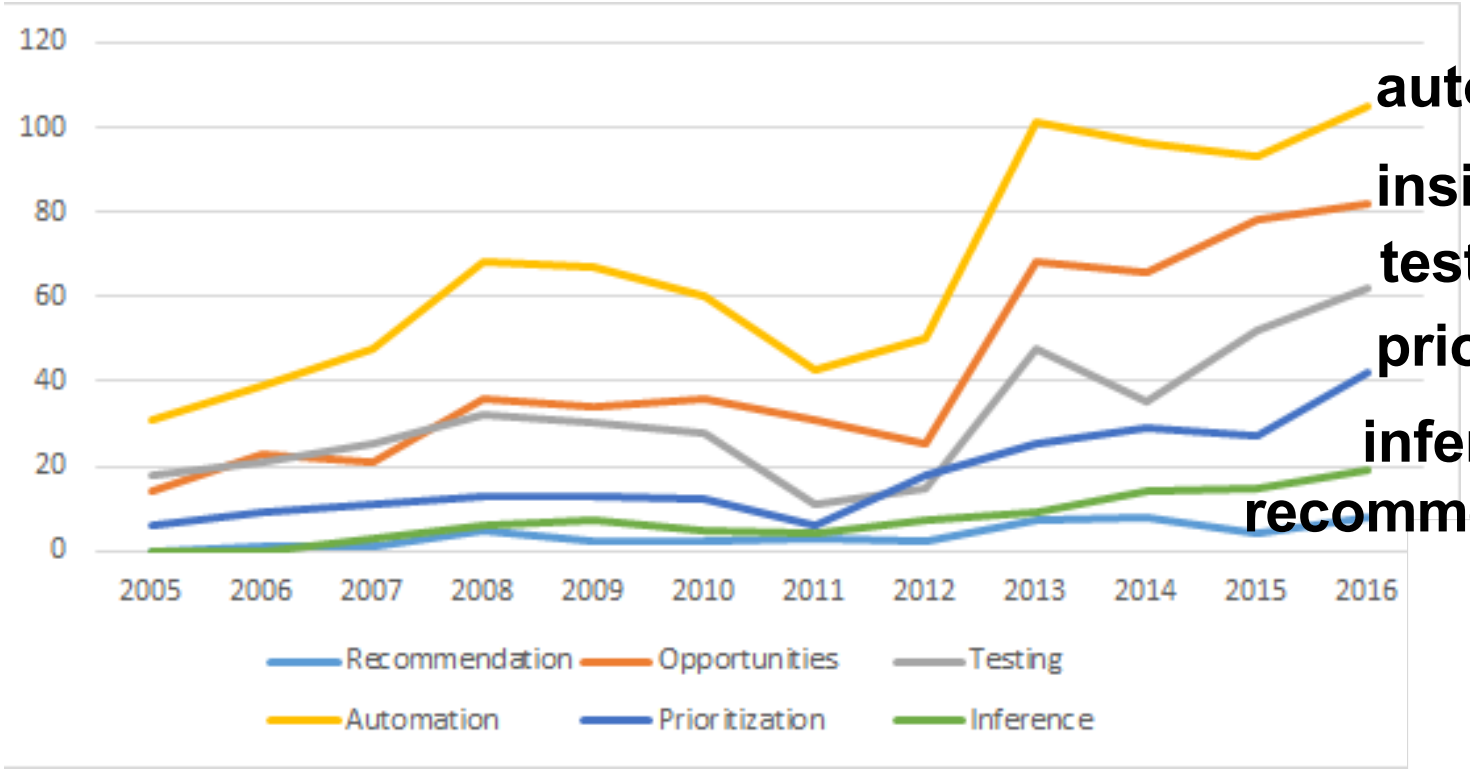
In the end we removed 397 papers

O1: To Grow, Welcome Outsiders, Champions from Other Communities



For PhD committee, invite 1-2 outsiders of your area

O2: To Grow, Expand Focus of Interest (the WHAT)



automating

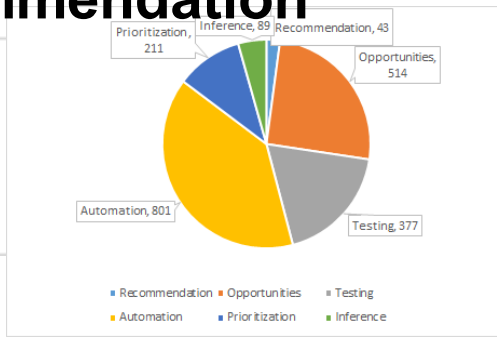
insight

testing

prioritization

inference

recommendation



Expand to meet the needs of new audiences

Our New Focus on Automation

Scalability 1.0: Refactoring to Design Patterns contain hundreds of lower-level refactoring steps [ICSE'16]
- 10x faster than state-of-the-art IDE refactorings

Scalability 2.0: Ultra-large scale refactoring for codebases of Hundreds of Millions LOC (e.g., Google Microsoft scale)
- whole-program analysis is not feasible

The next generation of global, distributed refactoring [ICSE19]
- MapReduce on the cloud: scalable, safe, useful

Our New Work on Inferring Refactorings

RefactoringMiner: commit-based detection **[ICSE'18]**

- **No similarity thresholds**
- **High accuracy**: 98% precision, 87% recall
- **Ultra-fast**: 58ms on median per commit
- **Better than previous state-of-the-art**: +22% precision, 7x faster

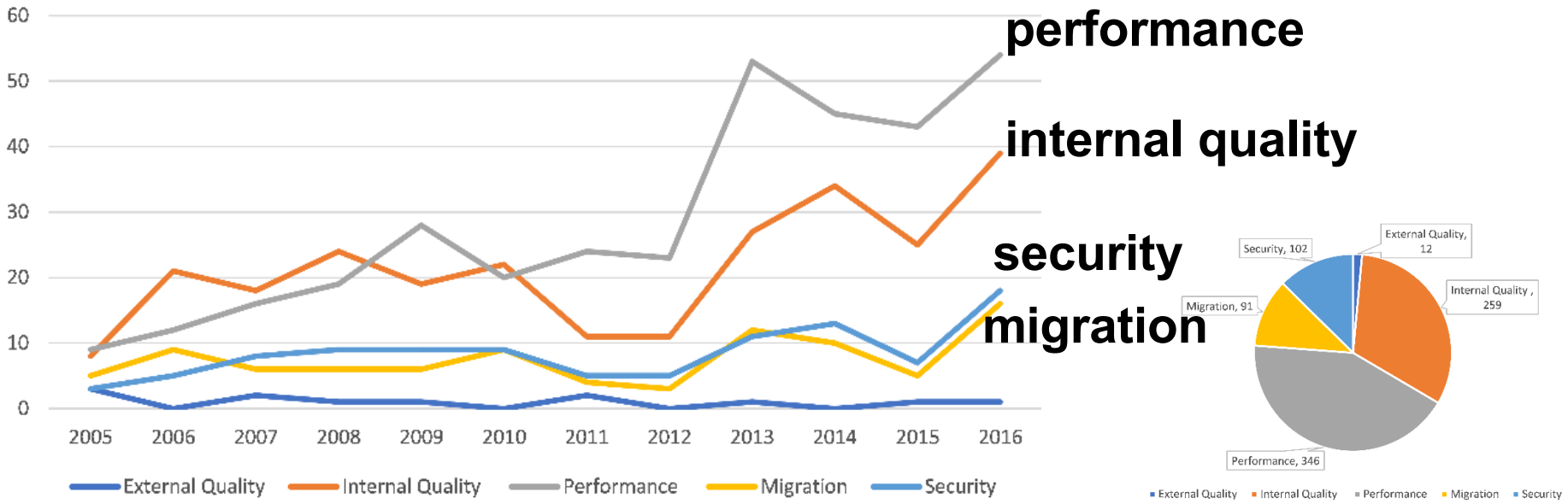
Largest and least biased refactoring oracle up to date

- **3188** true refactoring instances, **185** open-source projects

Enabling other researchers:

- **21 papers by others, within 1 year of our release**

O3: To Grow, Expand Objectives (the WHY)

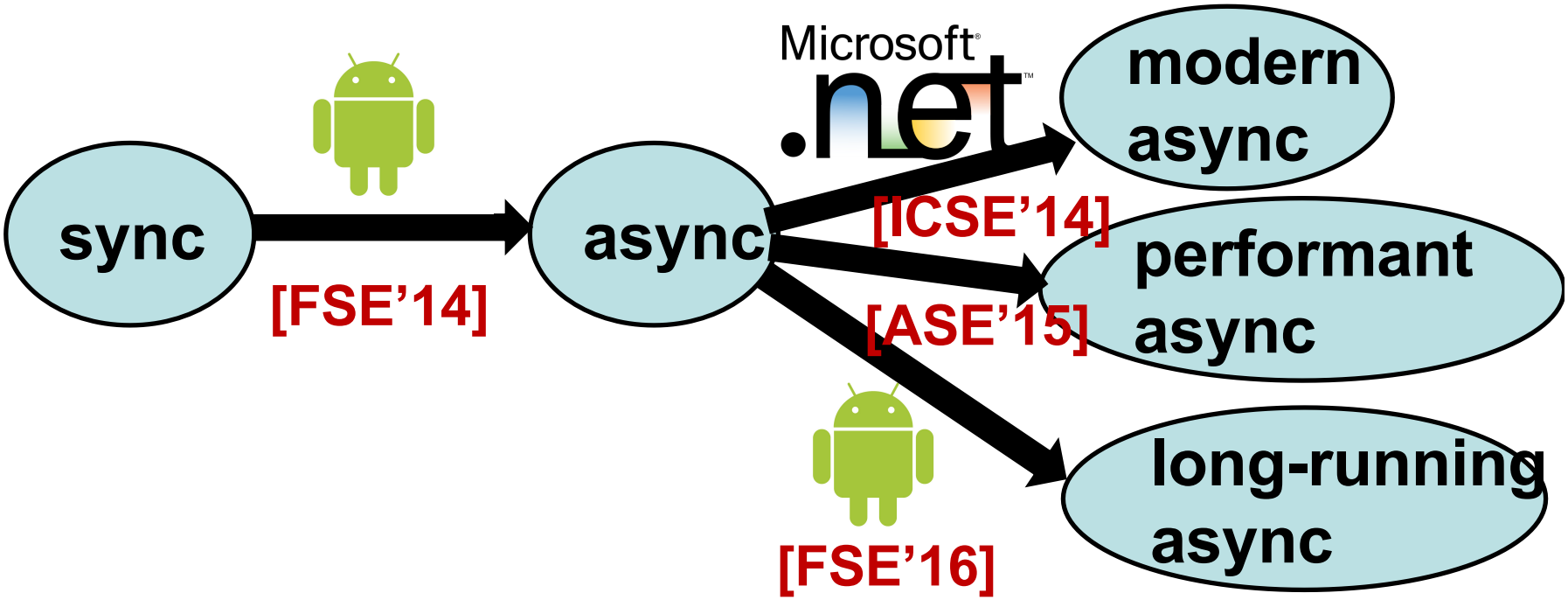


Expand Objectives: new refactoring research is to improve performance, security, migration (beyond internal quality)

Overview of Our Refactorings for Asynchronous Programming for Mobile Apps

Slow operations freeze mobile apps and frustrate users
- 75% of performance bugs in Android [Li et al., ICSE'14]

Culprit: long running operations running in the main UI thread
Solution: refactoring for asynchronous execution



Overview of Our Refactorings for Parallelism

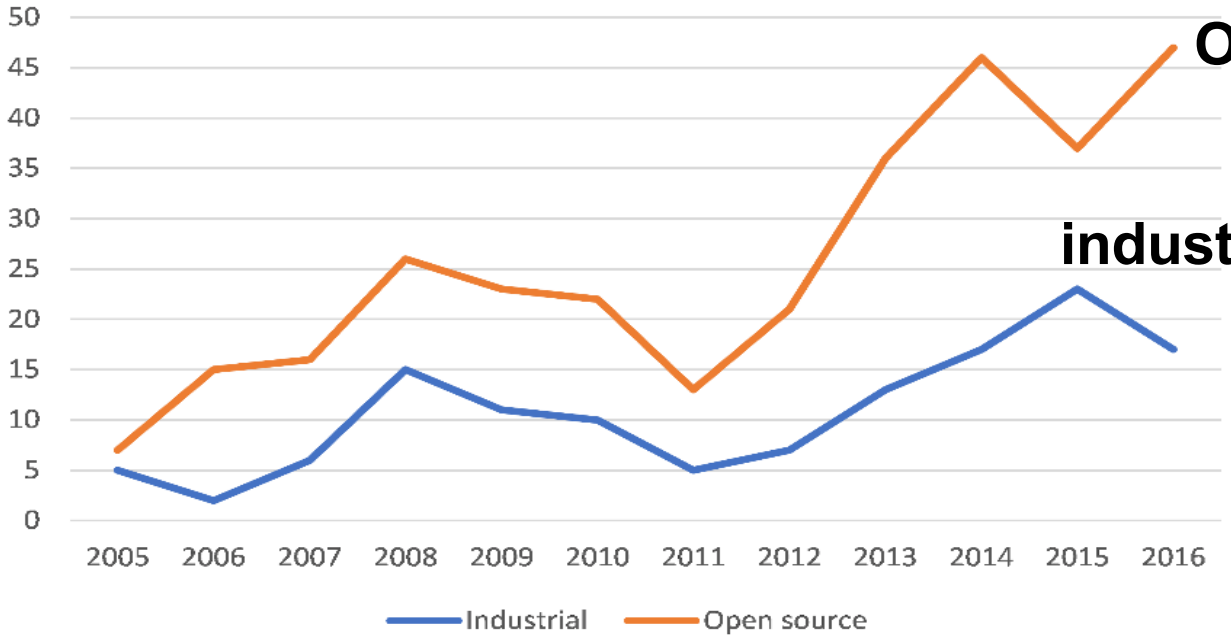
Refactorings for thread-safety
- make class immutable [ICSE'11]
- convert to Atomic* classes [ICSE'09]
- use concurrent collections [ICSE'09]
- infer region annotations [ASE'09]
- atomic check-then-act operations [ICST'13]

Interprocedural analyses:
- control, data-flow
- points-to
- constraint-based

Refactorings for throughput
- parallel recursive divide-and-conquer [ICSE'09]
- loop parallelism via ParallelArray [OOPSLA'10]
- loop parallelism via lambda functional operators [FSE'13]

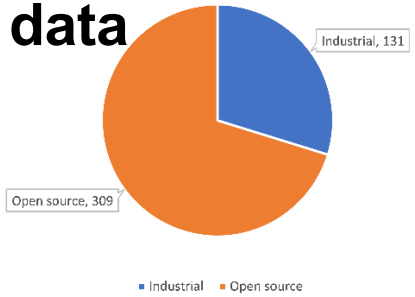
Refactorings for scalability
- Atomic*, concurrent collections [ICSE'09]

O4: To Increase Practical Impact, Go the Extra Mile



Open-source data

industrial data



Industrial collaboration levels:

- **surveys with practitioners**
- **tool validated on industrial codebase**
- **tool licensed to industry, adopted in products**

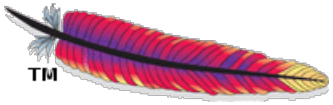
"There are no traffic jams on the extra mile"

What is Your Dream?
Mine is Practical Impact on SW Development

Automating

-ship with official

- hundreds of accepted patches



- first open-source refactoring

Refactoring

Inferring

- used at 

- dozen labs

founded Workshop on Refactoring Tools, HotSwUp, Dagstuhl S.

Understanding

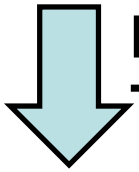
- shaped APIs in Java and .NET official concurency libraries

-learnparallelism.net
150,000+ visitors

Testing

Big Growth of the Field: Expanding Definition

"A change made to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behaviour" – M. Fowler '99



Expanded Focus, Objectives, Techniques

"Automation/insight/testing/prioritization of changes to the artifacts of software to improve non-functional requirements and without changing its proper, intended behaviour" – Danny Dig '19

Communities that grow are going to be more accepting of new ideas

Big Growth Enabled by Community Engineering

Industry champions: Martin Fowler, Kent Beck, Ward Cunningham, Joshua Kerievsky, Michael Feathers, Uncle Bob

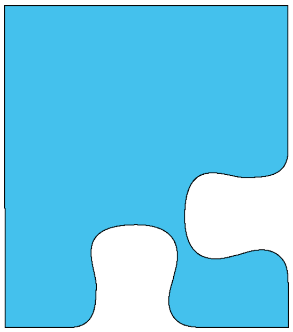
Complementary skills: tool builders, paper writers, curators

Mindset for industrial collaboration and adoption

Shared platform:
- **Eclipse (Erich Gamma + Frank Tip), analysis frameworks**

Community: 10 Refactoring Workshops, 1 Dagstuhl
- **first workshop in 2007, 50+ participants, 32 posters**
- **invited all major IDE providers**
- **growing new leaders**

Our New Refactoring for ML Code



90% of software costs due to software evolution

ML algorithms are also code: they need to evolve

Understand ML evolution through formative studies
- quantitative (static analysis) and qualitative

Automate transformations:
- retrofit ML: from deterministic to probabilistic
- model transformations
    - e.g., Eager to Static in TensorFlow
    - evolve model when data changes, without retraining

My Most Important Investment

Michael Hilton (PhD'17, now faculty at CMU)
Semih Okur (PhD'16, now at Microsoft)
Yu Lin (PhD'15, now at Google)
Stas Negara (PhD '13, now at Google)
Ameya Ketkar (PhD)
Malinda Dilhara (PhD)
Tom Dickens (PhD)
Sruti Srinivasa (PhD)
Shane McKane (MS'17, now at Intel)
Mihai Codoban (MS '15, now at Microsoft)
Kendall Bailey (MS '15, now at Intel)
Cosmin Radoi (MS '13, now PhD student UIUC)
Sandro Badame (MS '12, now at Google)
Fredrik Kjolstad (MS 2011, now PhD student MIT)
Binh Le (MS 2009, SW developer)
Can Comertoglu (MS 2009, now at Microsoft)

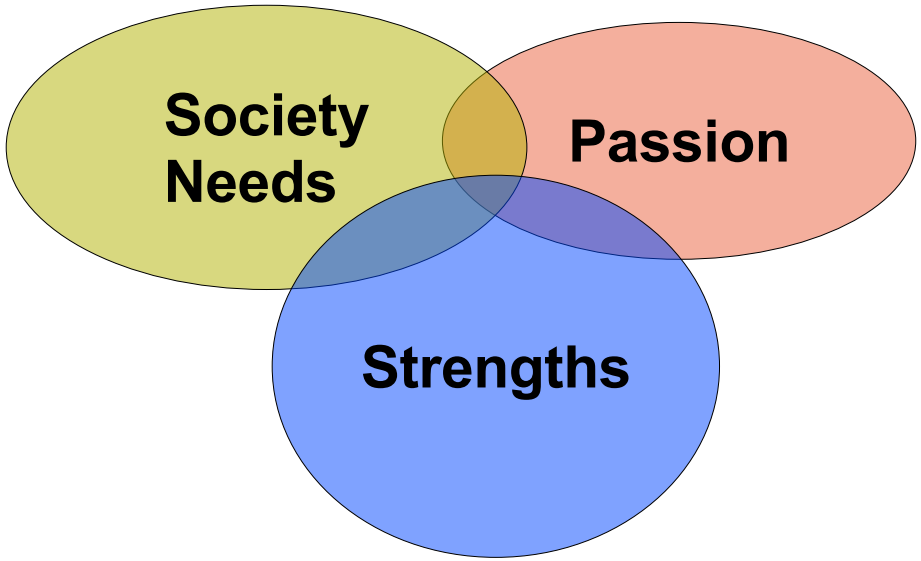
Jacob Lewis (Summer'16 – '17)
Jonathan Harijanto (Summer'16 –'17)
Lily Mast (Summer'15)
Elias Rademacher (Summer'15 - current)
Nicholas Nelson (Summer 2014-15)
Sean McDonald (Summer'14 –Fall'15)
Hugh McDonald (Summer'14 – Fall'15)
Alexandria Shearer (Summer'12)
Kyle Doren (Summer'12)
Lyle Franklin (UIUC, Summer'12)
Alex Gyori (UIUC, Summer'12)
Yuwei Chen (UIUC, Spring 2012)
Anda Bereckzy (UIUC, Fall'11-Spring'12)
Alex Sikora (UIUC, Fall'11)
Jack Ma (UIUC, Summer'11)
Lorand Szacaks (UIUC, Summer'11)
Caius Brindescu (UIUC, Summer'11)
Mihai Codoban (UIUC, Summer '11)
Mihai Tarce (UIUC, Summer'09)
Cosmin Radoi (UIUC, Summer'09)
John Marrero (MIT, Spring'08 – Summer'08)

Reflections and Lessons I am Learning

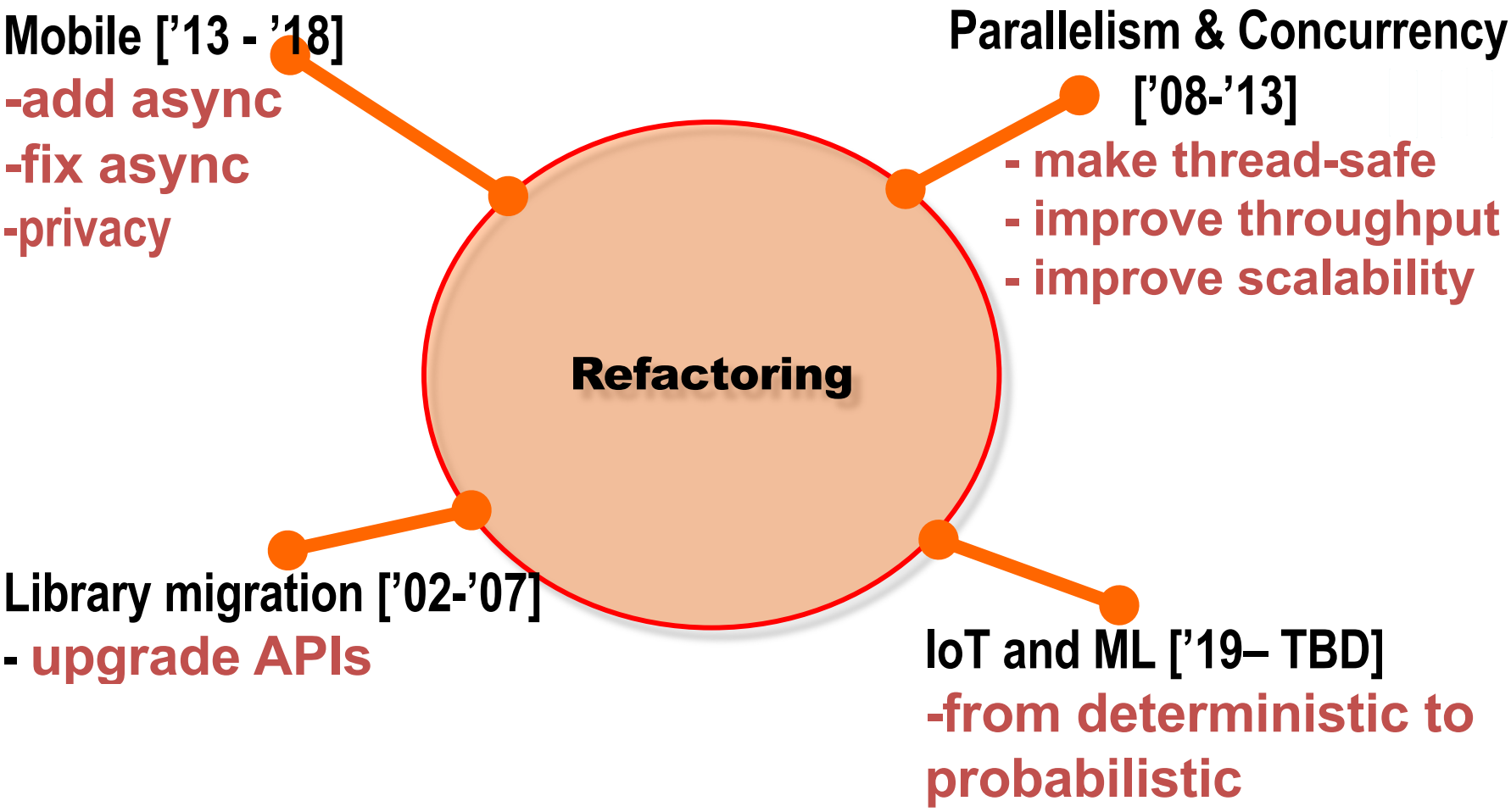
L1: Focus on the Students

"Focus on the students, since graduating great students means you'll produce great research, while focusing on the research may or may not produce great students" **-- the late David Notkin**

Helping students find their voice



L2: Work in Your Strength Zone but Reinvent Yourself



Principles for changing between different programming models

L3: Proactively Look for Opportunities, but Be Flexible

Expected Company	Actual Company	Expected Target	Actual Target
IBM	ORACLE	Lambda Expressions	Lambda Expressions
Google	Google	Async Programming	Type migration at scale

L4: To Grow Others, Grow Yourself

Do you have a plan for your personal growth?
How do you get better at what you do?
How do you improve your relationships?
How do you hire great people?
How do you mentor and grow them into tomorrow's tech leaders?
How do you prioritize the important over the urgent?