Summary & Impressions

Carly Lebeuf - Matthieu Foucault

Foundations of Software Engineering (FSE) Schedule

- Keynote Presentations (3)
  - **Margaret Burnett** - “Womenomics” and Gender Inclusive Software: What SE Need to Know
  - **James Herbsleb** - Building a Socio-Technical Theory of Coordination: Why and How
  - **Daniel Jackson & Mandana Vaziri** - Correct or Usable? The Limits of Traditional Verification
- Visions Presentations (2)
- Panel: The State of Software Engineering Research
  - Lionel Briand, Prem Devanbu, Peri Tarr, Laurie Williams, Tao Xie, Margaret-Anne Storey (mod.)
- Showcase of Software Engineering Best Practices
- Breakout Sessions (20)
- Collocated Workshops (8)

Proceedings can be found: http://dl.acm.org/citation.cfm?id=2950290&preflayout=flat

FSE Sessions

- Specification
- **HCI and Process**
- Bug Detection and Debugging
- Security and Privacy
- Adaptation and Change
- **API Mining and Usage**
- Verification
- **Requirements and Models**
- Android
- Static Analysis

- **Recommendation**
- Test Coverage
- Program Analysis
- Build and Configuration
- Code Search and Similarity
- Program Repair
- Development Environments
- Concurrency
- **Open-Source**
- Test Generation

Social Software Engineering (SSE) Workshop

Should We Take a Human-Centric View of Software Engineering by Adopting a Socio-Technical Perspective?

- **Jim Herbsleb**, Carnegie Mellon University, USA

The Rise and Fall of Developer Online Communities.

- **Chris Parnin**, NC State University, USA

Lessons in Social Coding: Software Analytics in the Age of GitHub.

- **Bogdan Vasilescu**, Carnegie Mellon University, USA

Should We Take a Human-Centric View of Software Engineering by Adopting a Socio-Technical Perspective?

Jim Herbsleb (http://sse-ws.github.io/FSE-Soc-Soft-2016-v6-dist.pdf)

- What Are the Building Materials for Software?
  - Church-Turing Thesis (Jim's paraphrase): Any Turing-complete machine can compute anything that is computable.
  - Implies that code running on any computer can (theoretically) fulfill any (computable) functional requirements.
- What Is the Problem?
  - Within the space of what is computable, limitations come from our own limited capacities
  - What can we understand?
  - What languages, abstractions, algorithms, and data structures can we dream up?
  - What are our limitations and how can we compensate for them?
  - How can we act together in a coordinated way?

Should We Take a Human-Centric View of SE...

Jim Herbsleb (http://sse-ws.github.io/FSE-Soc-Soft-2016-v6-dist.pdf)

Two Frameworks and an Example

- Transactive Memory Systems
  - Knowledge of “who knows what”
  - Develops through experience and collaboration
  - Facilitates adaptation to new situations or tasks
- Gatekeeper networks
  - Small number of people become information hubs
  - Connected to information sources inside and outside organization
  - People go to them with questions
- GitHub: Why so successful?
  - Provides means for humans to form and use social capabilities
  - Transactive Memory Systems: activity traces, profiles, consistent across repositories
  - Gatekeeper networks: Watching, starring, following, curating, “asynchronous mentoring”

Should We Take a Human-Centric View of SE...

Jim Herbsleb (http://sse-ws.github.io/FSE-Soc-Soft-2016-v6-dist.pdf)

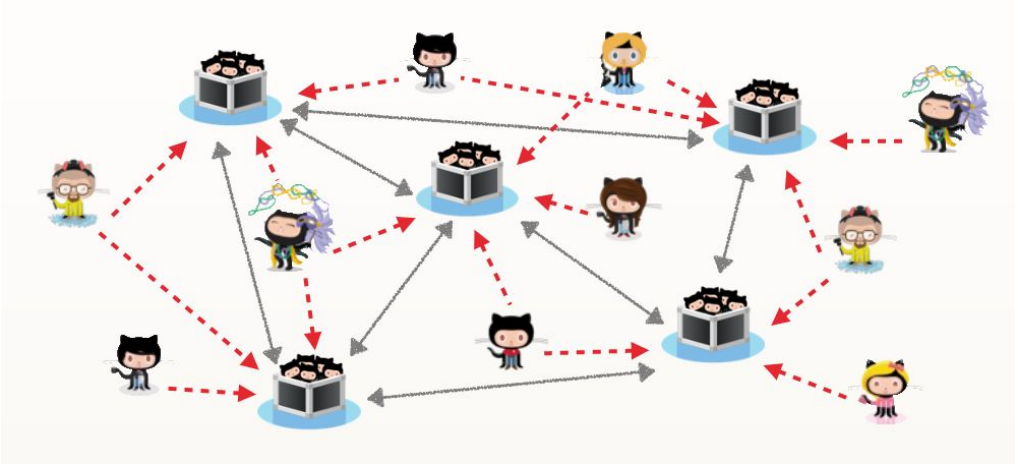
Takeaways...

- Psychology, sociology, etc. are a starting point to understand developers coordination
- Only moderately useful in current form
  - Stretched by complexity of environment, rapid change, capabilities of digital tools and materials
- We need a socio-technical perspective!

Lessons in Social Coding: Software Analytics in the Age of GitHub.

Bogdan Vasilescu

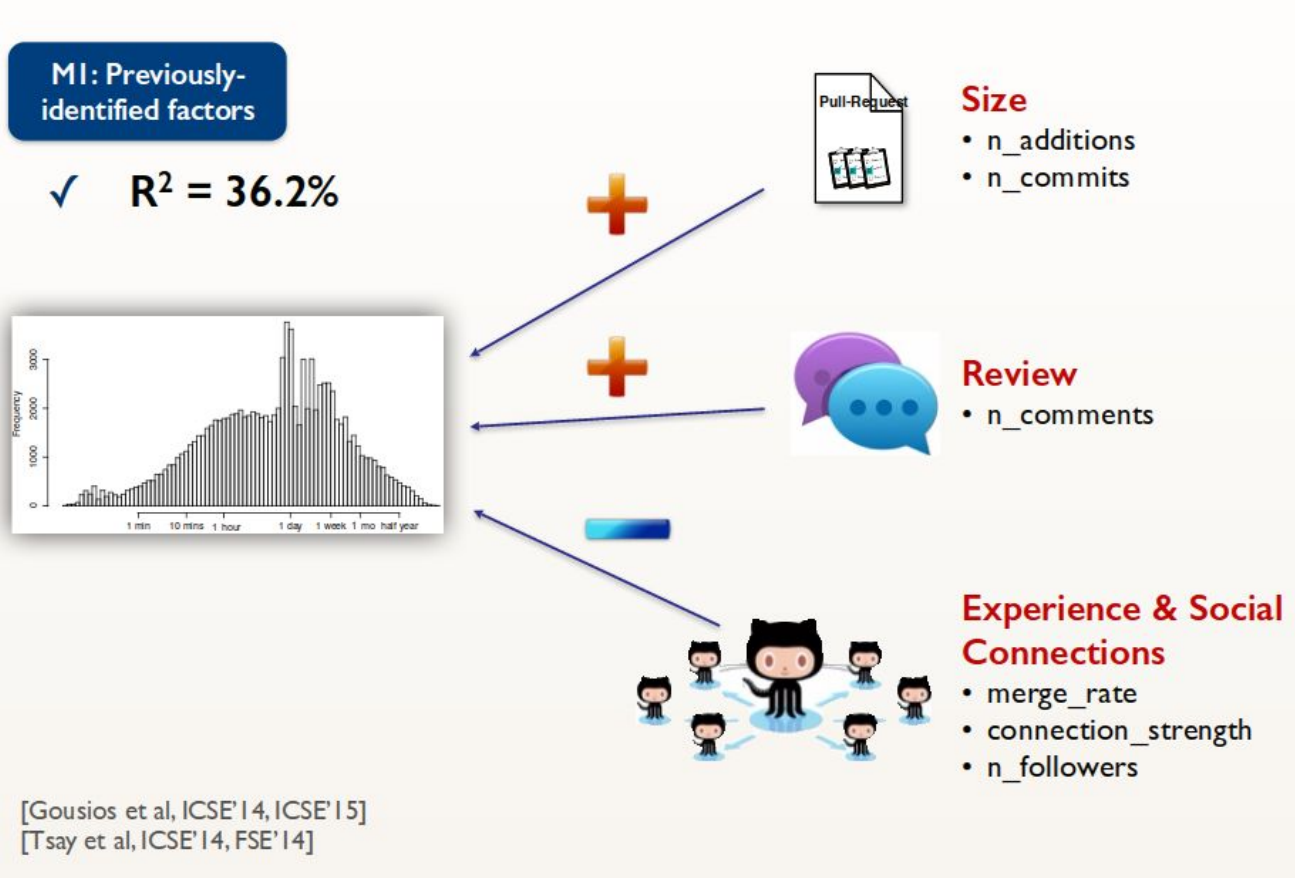
- Today's open-source development is happening in large, socially enabled ecosystems
- As practice is evolving, research should look at this new practice
- Two examples
  - Pull request evaluation time
  - Developer multitasking



SSE

Example 1: Pull Request Evaluation Time

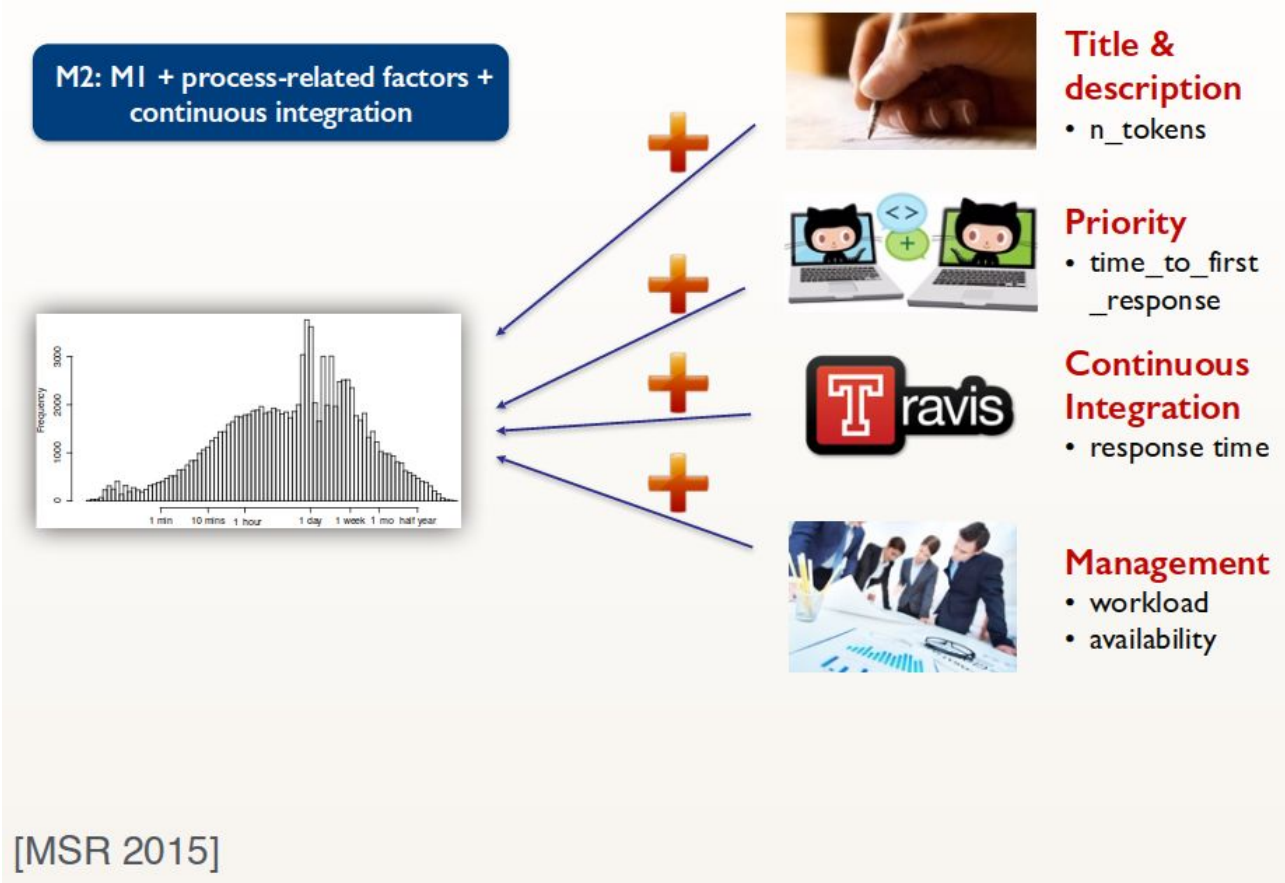
Bogdan Vasilescu



SSE

Example 1: Pull Request Evaluation Time

Bogdan Vasilescu



[MSR 2015]

SSE

Example 2: Multitasking and Performance

Bogdan Vasilescu

PROS

- **Fill downtime**

  Switch focus between projects to utilize time more efficiently

  (Adler and Benbunan-Fich, 2012)

- **Cross-fertilisation**

  Easier to work on other projects if knowledge is transferrable

  (Lindbeck and Snower, 2000)

CONS

- **Cognitive switching cost**

  Depends on interruption duration, complexity, moment

  (Altmann and Trafton, 2002)
  (Borst, Taatgen, van Rijn, 2015)

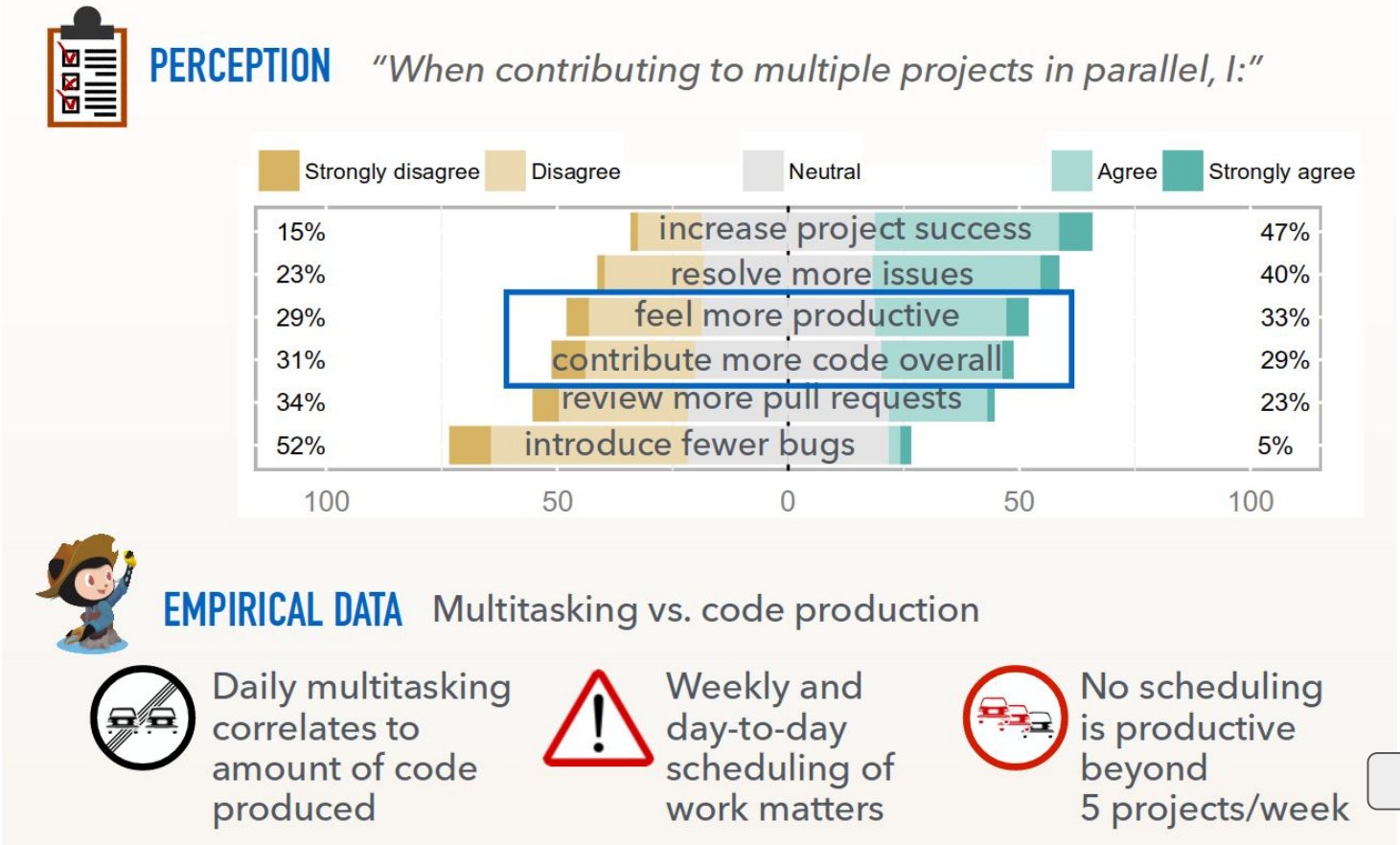
- **"Project overload"**

  Mental congestion when too much multitasking

  (Zika-Viktorsson, Sundstrom, Engwall, 2006)

Example 2: Multitasking and Performance

Bogdan Vasilescu



SSE

The Rise and Fall of Developer Online Communities

Chris Parnin (http://sse-ws.github.io/SSE-Parnin.pdf)

Traditional Documentation:

- Project (wrote by few, read by few) & API (wrote by few, read by many)

When developers are learning about API documentation (Microsoft Survey) they:

- Google (73.5%), IntelliSense (42.5%), Official Documentation (40.1%)

Study on JQuery API (2011)

- 1730 search results...

Search Result Type	Coverage	Mean Rank
code snippet site	8.7%	9
q&a	9.8%	9
forum	20.2%	8
official bug tracker	21.4%	3
mailing list entry	25.4%	7
official documentation	30.1%	3
official forum	37.0%	3
unofficial documentation	63.6%	6
stackoverflow	84.4%	6
blog post	**87.9%**	**5**
official API	99.4%	1

SSE

The Rise and Fall of Developer Online Communities

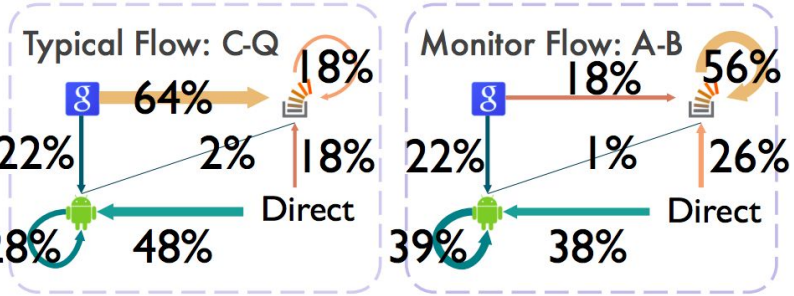
Chris Parnin (http://sse-ws.github.io/SSE-Parnin.pdf)

Crowd Documentation

- “Knowledge is created and curated by a mostly uncoordinated collective”
- An example of Peer Production

Is “Crowd Documentation” used?

- 1,316 days of developer browser history
- Consistent with the self-reported surveys



SSE

The Rise and Fall of Developer Online Communities

Chris Parnin (http://sse-ws.github.io/SSE-Parnin.pdf)

What makes Stack Overflow different?

- Traceability links, quick response times, high coverage (88% of Android API), correlated with usage, more examples, experts

The downfall of Stack Overflow…

- Takes a long time to get coverage (3 years to get 50% coverage on GWT)
- Limited topics covered (ex. accessibility)
- Gamification mechanisms: 60% of questions answered by 5% of users
- Participation: 21% of users are female, but only 5-7% contribute
- Barriers: fear, saturation, microaggressions,

Automated Community Repair:

- Repair bots (fix docs / warn), Community bots (monitor / pair up devs)

Paradise Unplugged: Identifying Barriers for Female Participation on Stack Overflow

D. Ford, J. Smith, P. Guo, C. Parnin (doi.org/10.1145/2950290.2950331)

Conducted **22 interviews** with female developers & a follow-up **survey (134 F, 1336 M)** to determine **barriers** that existed for contributing on Stack Overflow.

The following categories (3) of barriers (14) were found:

- “Muddy Lens Perspective” - unclear perception of how Stack Overflow works
- “Impersonal Interactions” - lack of connections / uncomfortable atmosphere
- “On-Ramp Roadblocks” - obstacles that undermined interest in posting

Some barriers (5) were found to be **significantly more problematic** for females.

Open Source

Paradise Unplugged: Identifying Barriers for Female Participation on Stack Overflow

D. Ford, J. Smith, P. Guo, C. Parnin (doi.org/10.1145/2950290.2950331)

Group	Barrier	Participant Count	Description
Muddy Lens Perspective	Awareness of Site Features	11	I feel I am simply unaware of and have not explored the more advanced features of the site.
	Nothing Left to Answer	10	I feel all the easy questions have already been answered, leaving only hard questions.
	Fear of Contributing to Clutter	9	I feel my question might just be a duplicate or unimportant question, so I refrain from posting.
	No "Good-Answer" Guarantee	7	When posting a question, I fear not getting a good answer.
	Perception of Slacking	4	I feel that I should not be spending time answering questions on Stack Overflow for my own personal benefit.
Impersonal Interactions	Fear of Negative Feedback	18	I fear my posts being harshly criticized by users on the site.
	Stranger Discomfort	9	I feel uncomfortable interacting with and relying on help from strangers online.
	Intimidating Community Size	9	I feel intimidated by the large community of users. I instead prefer connecting with a smaller and more intimate group.
	Posting is Hard, Friends are Easy	6	I feel the process of posting questions is too cumbersome compared to other resources such as asking friends for help.
On-Ramp Roadblocks	Abstraction Process	20	I feel my problems require too many dependencies or proprietary aspects for me to abstract away before having something I can ask to a general audience.
	Time Constraints	17	I feel making contributions on Stack Overflow requires more time than I have.
	Qualifications	13	I feel my expertise or answers would not be of any help to anyone else.
	Onboarding Hoops	9	I feel figuring out the unspoken social etiquette and community standards is too much work.
	Research Pressure	9	I feel discouraged by the amount of work I have to do to prove that I'm not asking a duplicated question.

Open Source

“Womenomics” and Gender-Inclusive Software: What Software Engineers Need to Know

Margaret Burnett (1:30-2:30 pm, January 6th, ECS 660)

User’s experiences with software from a gender perspective...

Introduced the **GenderMag Method**...

- Helps software developers / usability experts identify features that are not gender-inclusive
- **5-facets of gender differences:** motivations for use, information processing style, computer self-efficacy, attitude towards risk, willingness to explore / tinker
- **4 Personas** representing “archetypes” of user
  - A set of male / female personas to bring to life the 5-facets of gender differences
- **Cognitive Walkthrough** that explicitly uses the 5-facets of gender differences and the personas

Keynote

Disrupting Developer Productivity One Bot at a Time

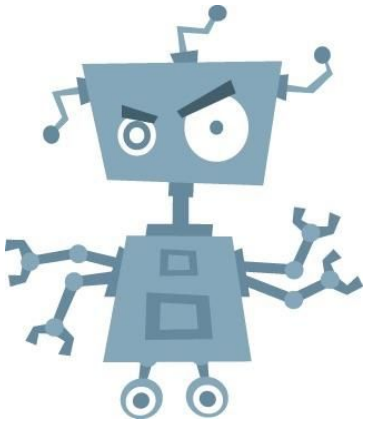
Margaret-Anne Storey & Alexey Zagalsky (doi.org/10.1145/2950290.2983989)

What is a bot?

- A bot is an application that performs **automated, repetitive, pre-defined** tasks
- Conduit between users and services, typically through a conversational UI
- “The operating system of the future isn’t Windows, but **conversation as a platform**” - Microsoft

The five proposed **dimensions of bots**…

- What do they **do**…
- How **intelligent**…
- How **autonomous**…
- How to **interact** with them…
- How they are **created**…



Visions

Disrupting Developer Productivity One Bot at a Time

Margaret-Anne Storey & Alexey Zagalsky (doi.org/10.1145/2950290.2983989)

Bots in Software Development…

- **Entertainment** bots
- **Code** bots
- **Test** bots
- **DevOps** bots
- **Support** bots
- **Document** bots

Productivity framework for bots…

- **Efficiency**: “do things faster”
  - Automate repetitive tasks
  - Help developers stay in the flow
- **Effectiveness**: “work towards goals”
  - Decision making
  - Team cognition, self / team regulation

What **risks** do we need to consider when using bots?

- Will bots change how **humans relate to one another**?
- What **ethical framework** should be used for bots?
- When don’t bot **interactions** work?

Visions

Designing for Dystopia: Software Engineering Research for the Post-Apocalypse

T. Barik, R. Pandita, J. Middleton, E. Murphy-Hill (doi.org/10.1145/2950290.2983986)

Software Engineers are generally optimistic, but this bias bolsters **unrealistic expectations** towards desirable outcomes

Explicitly **framing software engineering research with dystopias** may…

- mitigate optimism bias
- encourage more diverse, thought-provoking research directions

Explores the application of 3 dystopias in Software Engineering:

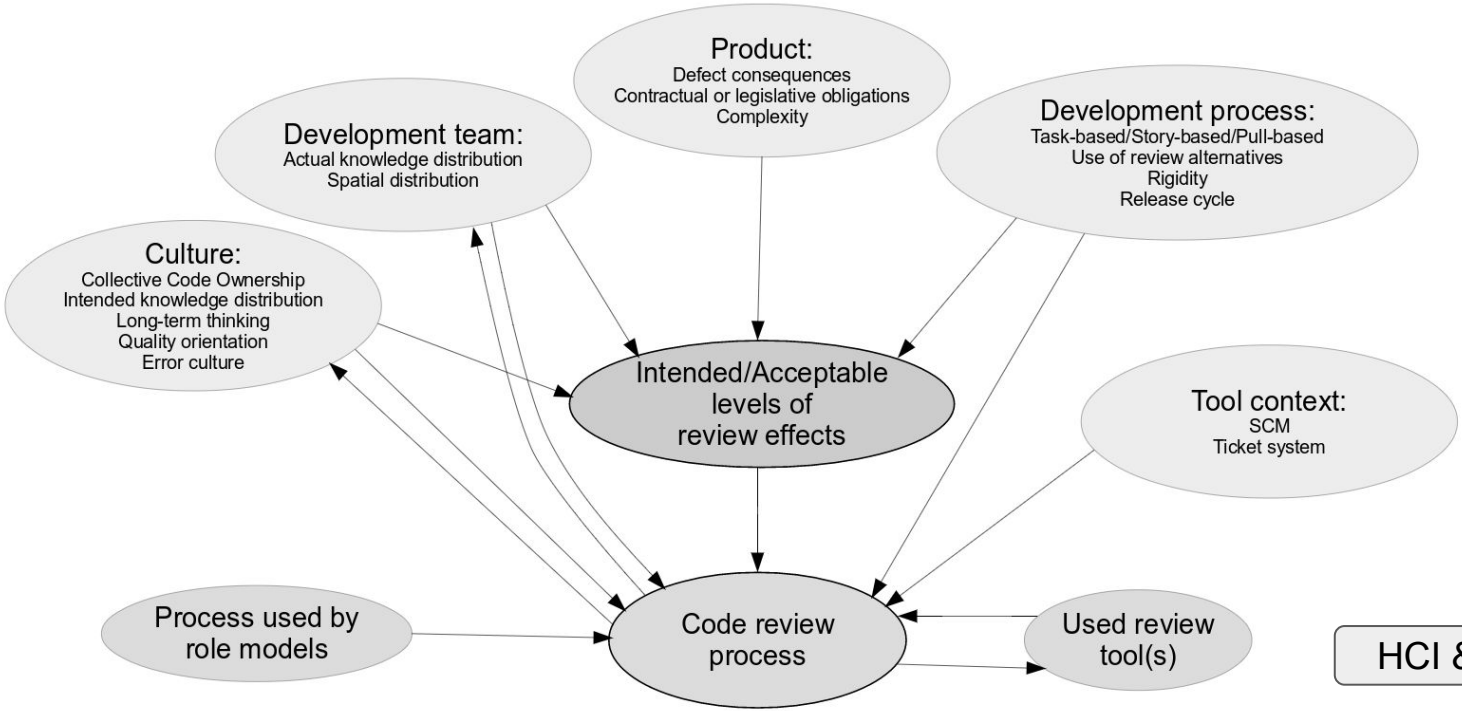
- **Battlestar Galactica**: skeptic of technology since it may be hackable
- **Fallout 3**: limited resources, new programs / patches are risky / costly
- **Children of Men**: support existing software, rather than building new software

Visions

Factors Influencing Code Review Process in Industry

T. Baum, O. Liskin, K. Niklas, K. Schneider (doi.org/10.1145/2950290.2950323)

- Investigate the adoption or non-adoption of code reviews
- Interviews of developers from 19 companies



HCI & Process

Why we refactor? Confessions of GitHub contributors

D. Silva, N. Tsantalis, M.T. Valente (doi.org/10.1145/2950290.2950305)

- Mainly driven by changes in requirements, not so much code smells resolution
- Motivations for Extract Method: reusability, introduction of alternative signature, improve readability, facilitate extension
- Main motivation for Move Class/Attribute/Method: conceptual relevance
- Refactorings remain manual half of the time
  - Inheritance-related refactoring tools are the less used (10% done automatically)
  - Renaming-related refactorings are the most trusted (over 50% done automatically)
- The IDE matters: IntelliJ users perform more refactorings than Eclipse users

Open Source

When should internal interfaces be promoted to public?

A. Hora, M. Valente, R. Robbes, N. Anquetil (doi.org/10.1145/2950290.2950306)

Software systems often have **public (stable) APIs** & **internal (unstable) APIs**

- Clients often use internal interfaces, causing **failures** when the APIs evolve
- API producers may promote internal interfaces to public
- There is currently **no way of detecting** internal interface promotion candidates

Conducted an empirical investigation on 5 Java systems:

- Promoted interfaces are domestically used by more packages, classes, commits and developers, and that they tend to attract newer clients over time
- Applied predictor to automatically detected 382 public interface candidates
- Public interface candidates interfaces were more likely to external clients

API Mining

How to break an API: cost negotiation and community values in three software ecosystems

C. Bogart, C. Kästner, J. Herbsleb, F.Thung (doi.org/10.1145/2950290.2950325)

- In Eclipse, you don’t
- In R, you reach to downstream developers
- In NPM, you use semantic versioning

HCI & Process

FSE Panel





Panelists: Lionel Briand, Prem Devanbu, Peri Tarr, Laurie Williams, Tao Xie
Moderator: Margaret-Anne Storey

Three questions were posed to the panel:

1. Do you believe our community as a whole is achieving the **right balance of science, engineering, and design** in our combined research efforts?
2. What new or existing **areas of research do you think our community should pay more attention to**?
3. Do you have novel suggestions for how we could **improve our research methods** to increase the impact of software engineering research in the near and distant future?

Recording: https://youtu.be/sE_jX92jJr8, Blog Post: www.margaretstorey.com

FSE / ESEC 2017 will be held in Paderborn, Germany

Call for papers deadline: February 27th, 2017

2016 Proceedings: http://dl.acm.org/citation.cfm?id=2950290&preflayout=flat