Carnegie Mellon University

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

Jim Herbsleb

Yes.

Agenda

- Evolution provides capabilities as users ***and builders***
- Modular capabilities
- Several useful theories of human capability
  - Transactive memory systems
  - Social network analysis
  - Socio-technical congruence can provide a framework
- Software engineering is both a technical and a behavioral science
  - What can we do with this?

What Is Engineering?

- Creating cost-effective solutions
  - Engineering is not just about solving problems; it is about solving problems with economical use of all resources.
- to practical problems
  - Engineering deals with practical problems whose solutions matter to people outside the engineering domain-the customers.
- by applying scientific knowledge
  - Engineering solves problems in a particular way: by applying science, mathematics, and design analysis.
- to building things
  - Engineering emphasizes the solutions, which are usually tangible artifacts.
- in the service of mankind.
  - Engineering not only serves the immediate customer, but it also develops technology and expertise that will support the society.

From Shaw, M. Prospects for an engineering discipline of software. *IEEE Software*, 7, 6 (1990), 15-24.

What Is Engineering?

- Creating cost-effective solutions
  - Engineering is not just about solving problems; it is about solving problems with economical use of all resources.
- to practical problems
  - Engineering deals with practical problems whose solutions matter
- by applying scientific knowledge
  - Engineering solves problems in a particular way: by applying science, mathematics, and design analysis.
- to building things
  - Engineering emphasizes the solutions, which are usually tangible artifacts.
- in the service of mankind.
  - Engineering not only serves the immediate customer, but it also develops technology and expertise that will support the society.

From Shaw, M. Prospects for an engineering discipline of software. *IEEE Software*, 7, 6 (1990), 15-24.

Bird

Microsoftus Christianus


Bird as user


Bird as builder

What does this tell us about a socio-technical perspective?

- To have a science and improve the practice of avian architecture, study
  - Properties of the materials and compositions
  - Properties of birds as users
  - Properties of birds as designers/builders/teams
- How effective would we be at improving avian architecture if we ignored social properties of birds as builders?

Mental Equipment: Evolution

- Adapted to hunter/gatherer way of life
  - Mental and physical capabilities
  - Evolution is a slow process
  - We did not change much in last 12,000 years (since the agricultural revolution*)
- We need to use mental equipment suited to simple hunter/gatherer life to design and build software

*Harari, Y. N. *Sapiens: A brief history of humankind* Random House, 2014.

Specific Cognitive Modules

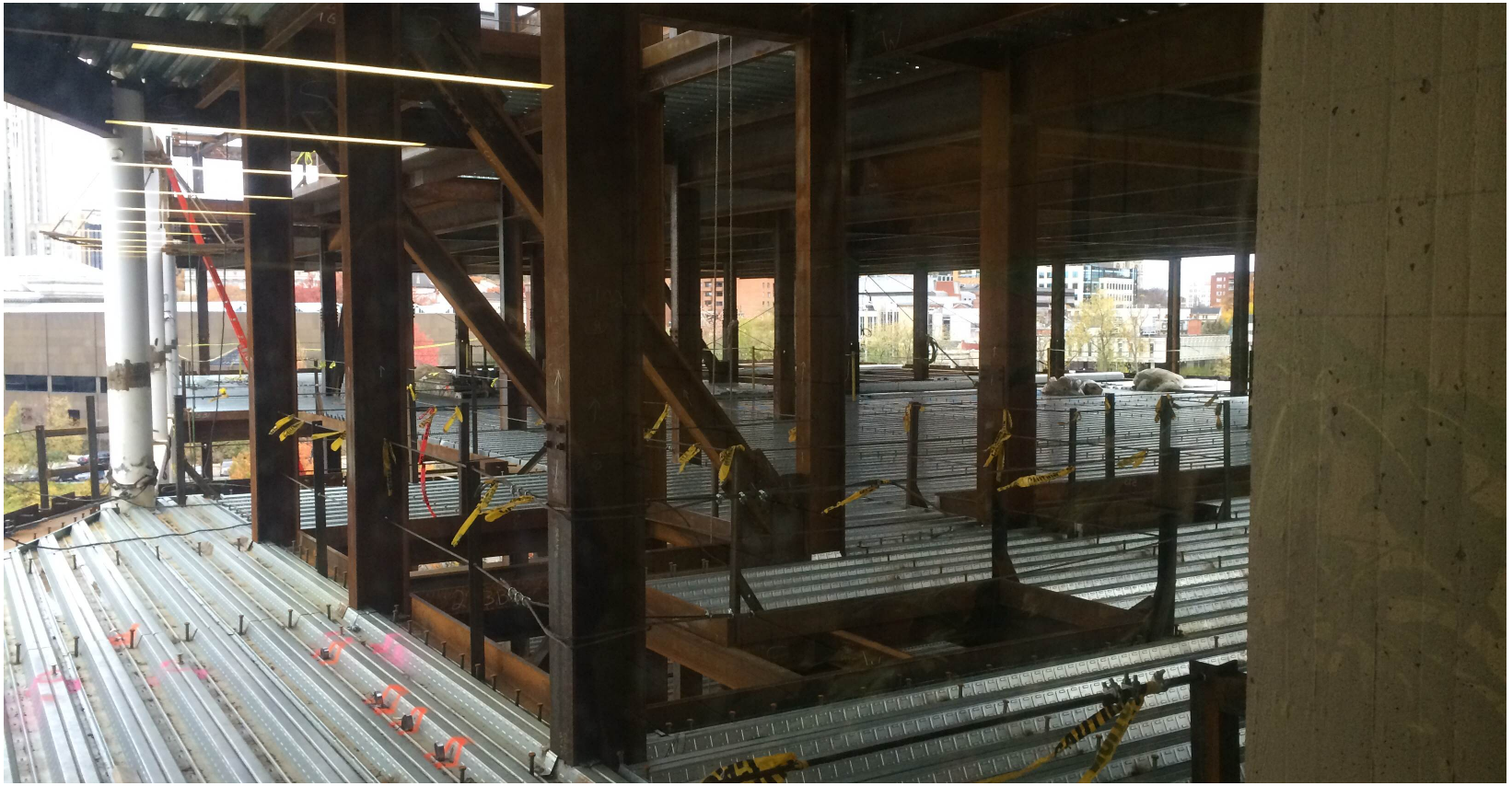
- Examples:
- Acquiring natural language
  - Chomsky: our brain has a built-in language acquisition device (LAD)
- Visually interpreting 3D space
  - The best terrain modeling programs are vastly inferior
- “Theory of mind”
  - Interpret and predict behavior of others based on inferred beliefs and desires

"The peculiar properties of the artifact lie on the interface between the natural laws within it and the natural laws without."

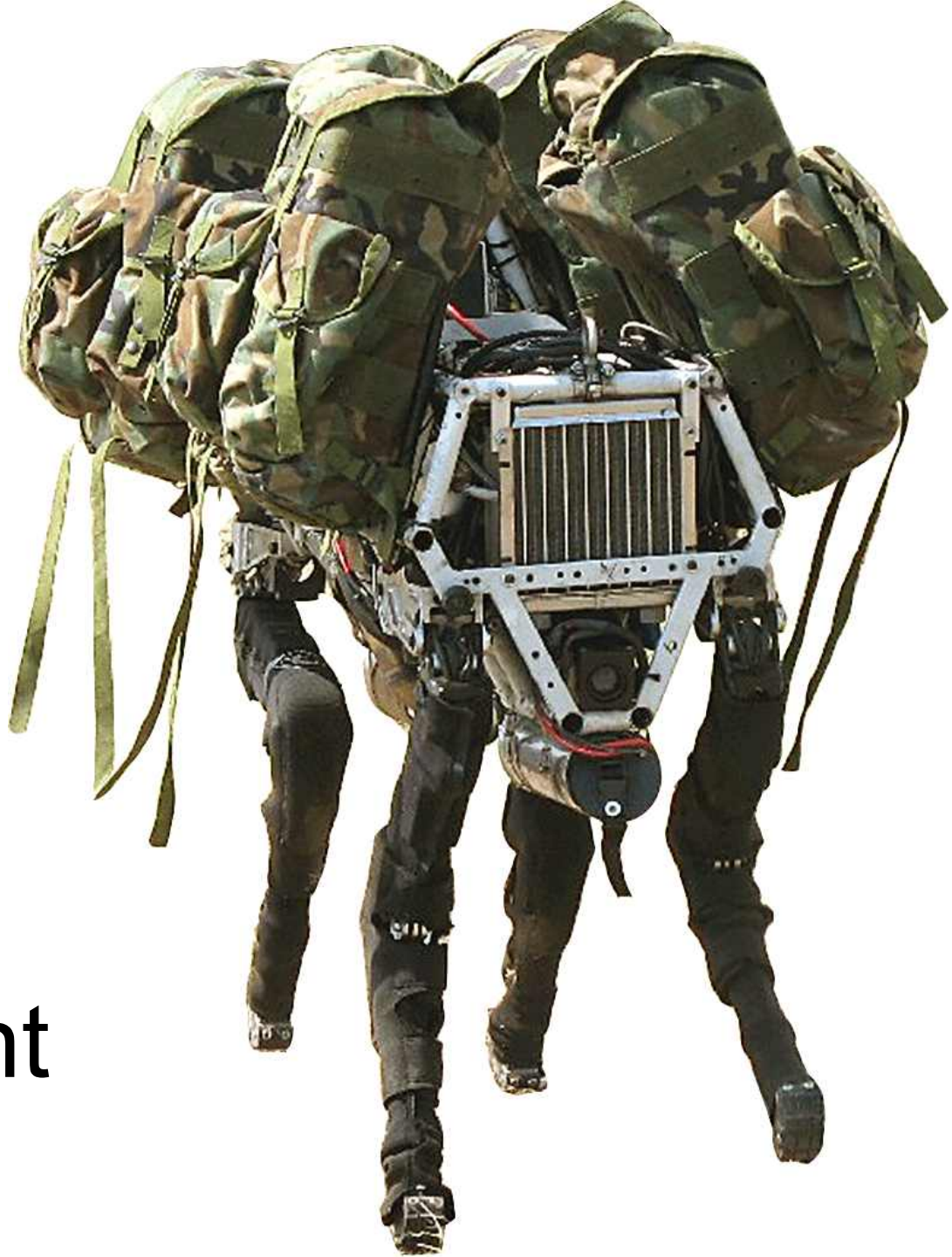
Simon, Herbert A. *The Sciences of the Artificial* (MIT Press) (p. 112-113). The MIT Press.

Outer Environment

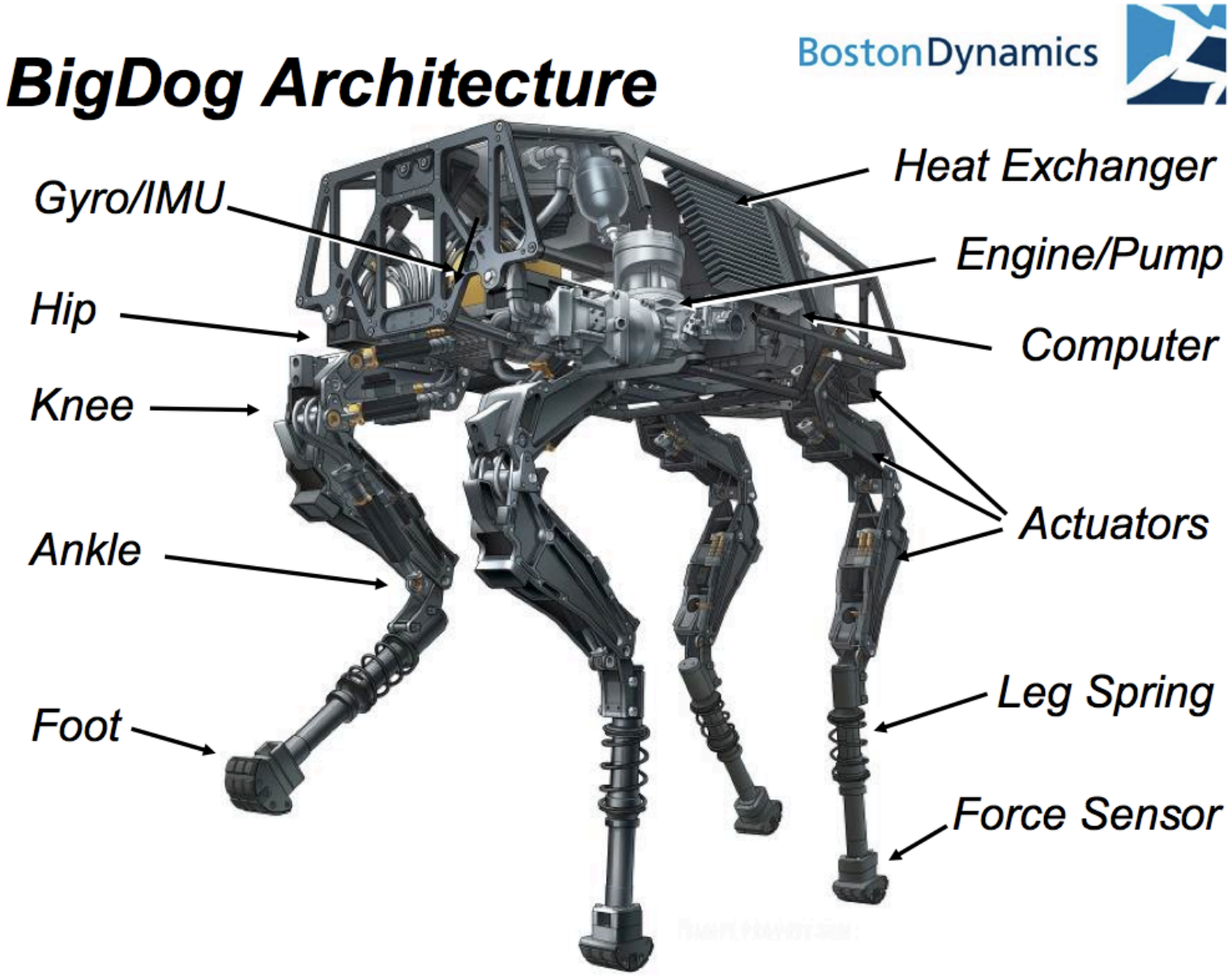
Inner Environment



Outer Environment



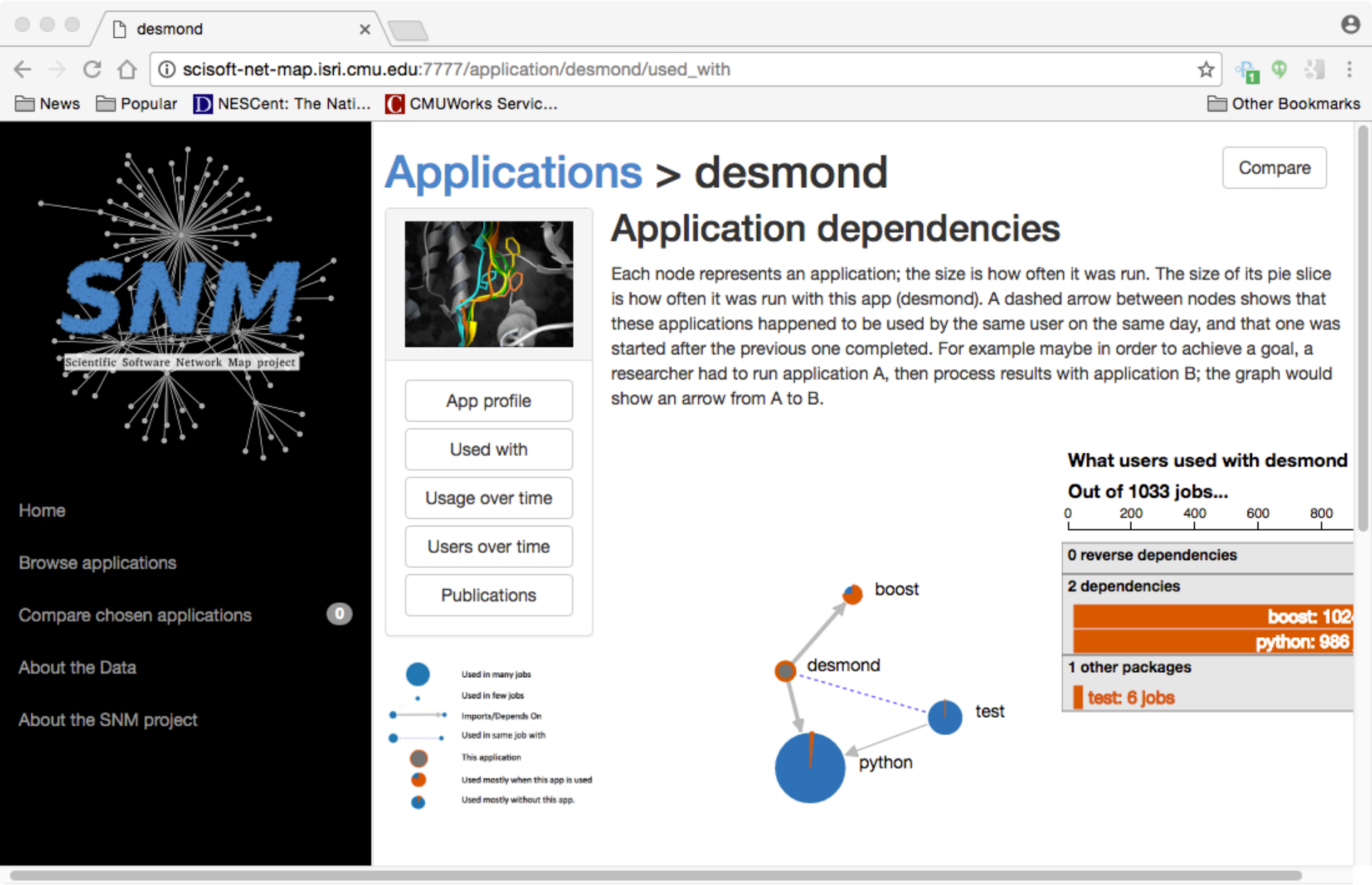
Inner Environment



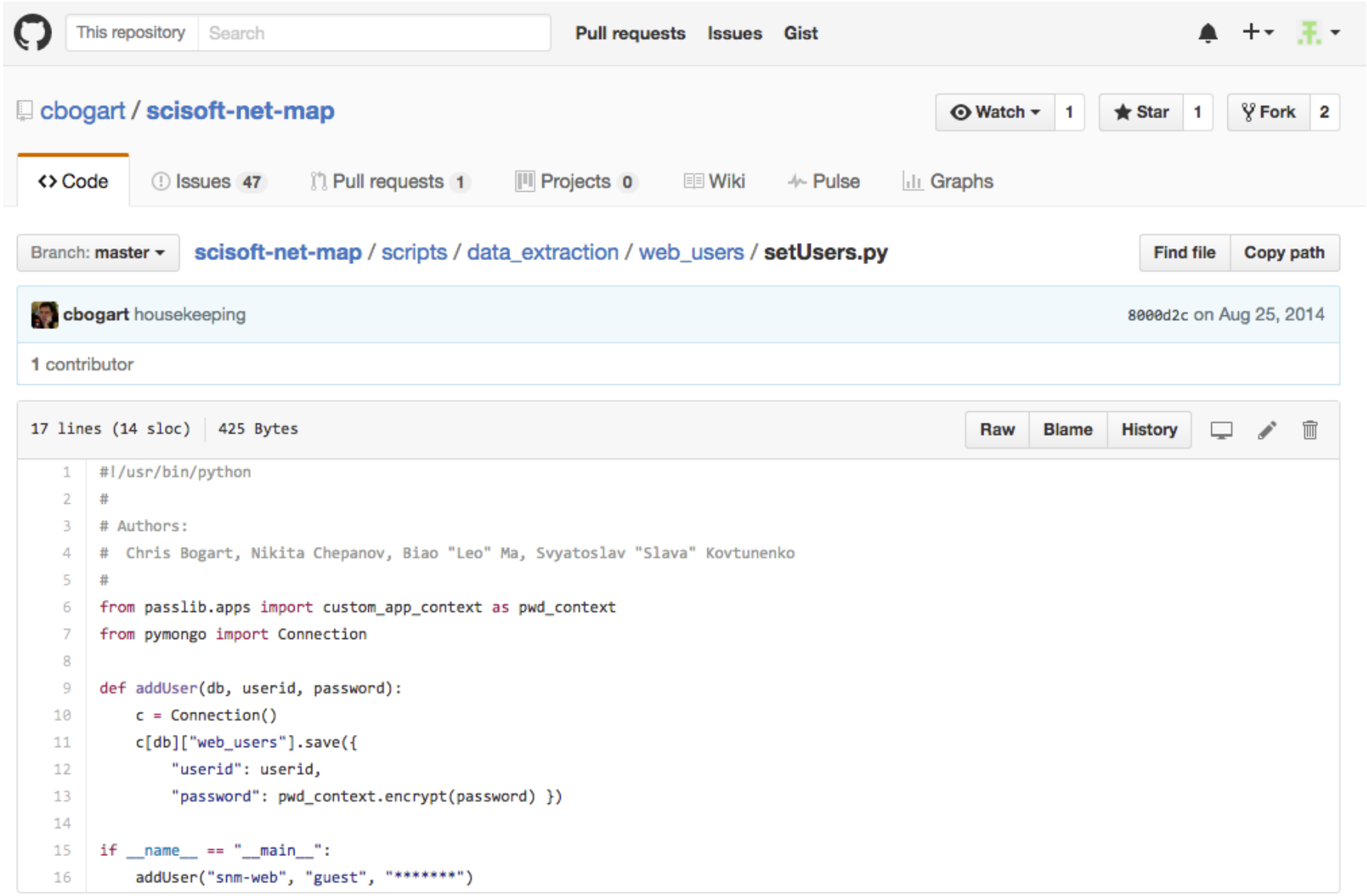
Many Engineering Challenges Have to do with Physical Stuff

- Strength of structural pieces
- Power consumption and output of a motor
- Power storage
- Sensitivity of sensors
- Etc., etc.
- The science they need – properties of physical and electronic components and compositions

Outer Environment



Inner Environment



What Are the Building Materials for Software?

- Church-Turing Thesis (my 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.
- Analogy
  - Can you (theoretically) build any building out of any construction material?
  - Can you build it in any order, e.g., from roof down, etc.?
  - This is the kind of freedom a “perfect” 3D printer would give us

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?

Two Examples and a Framework

- Transactive memory systems
- Gatekeepers and social networks
- Socio-technical coordination
  - Software project as distributed constraint satisfaction problem
  - Solved by executing social algorithms

Transactive Memory Systems (TMS)

- Group level phenomenon
- Arises naturally
- Specialization + index
  - People take responsibility for group knowledge and memory in some area
  - Everyone develops an index of “who knows what”
  - Origins in people watching each other work
  - Develop under wide range of conditions, but some are more favorable than others
- Very powerful impacts on how well groups function

TMS: Benefits and Conditions

- Specialization gives better performance
- Better coordination, agree on responsibilities
- Facilitates adaptation to new situations or tasks
- Facilitates creativity
- Developers under right conditions
  - Observe each other working
  - Communication
- Complex and socially embedded: hard to observe

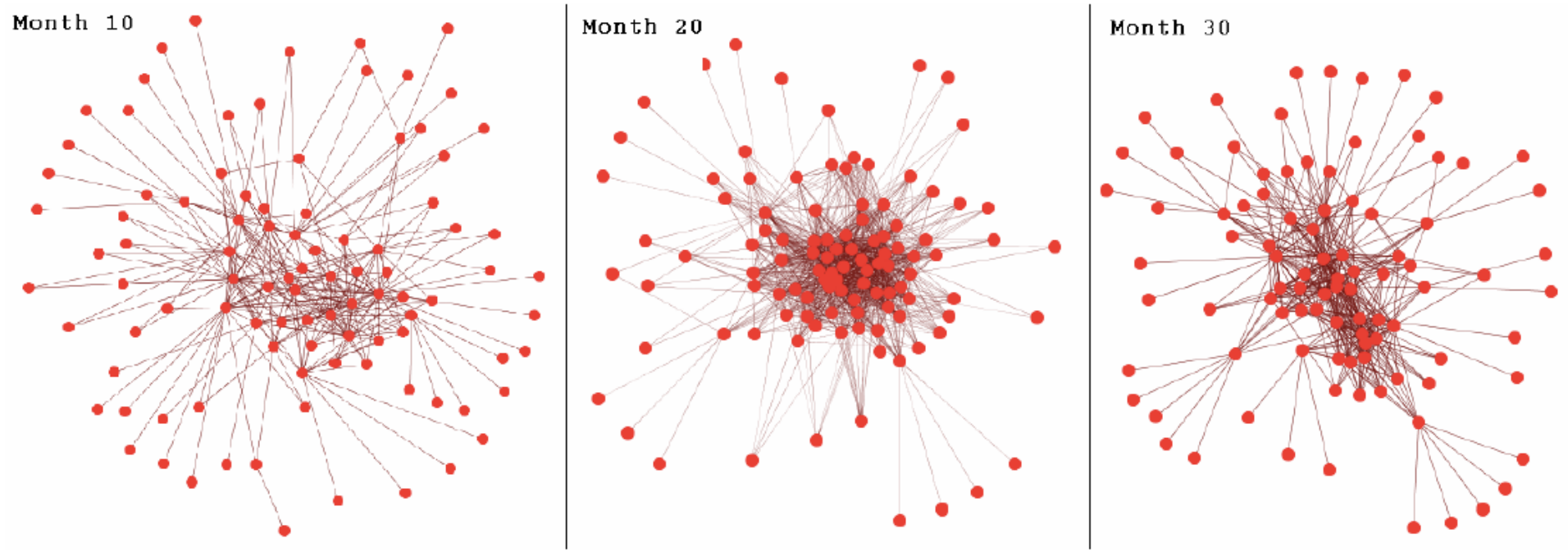
Argote, L. and Ren, Y. Transactive memory systems: A microfoundation of dynamic capabilities. *Journal of Management Studies*, 49, 8 (2012), 1375-1382.

Gatekeepers and Social Networks

- Small number of trusted people become information hubs
- Connected to information sources inside and outside organization
- People go to them with questions
- They form their own network, know each other's expertise

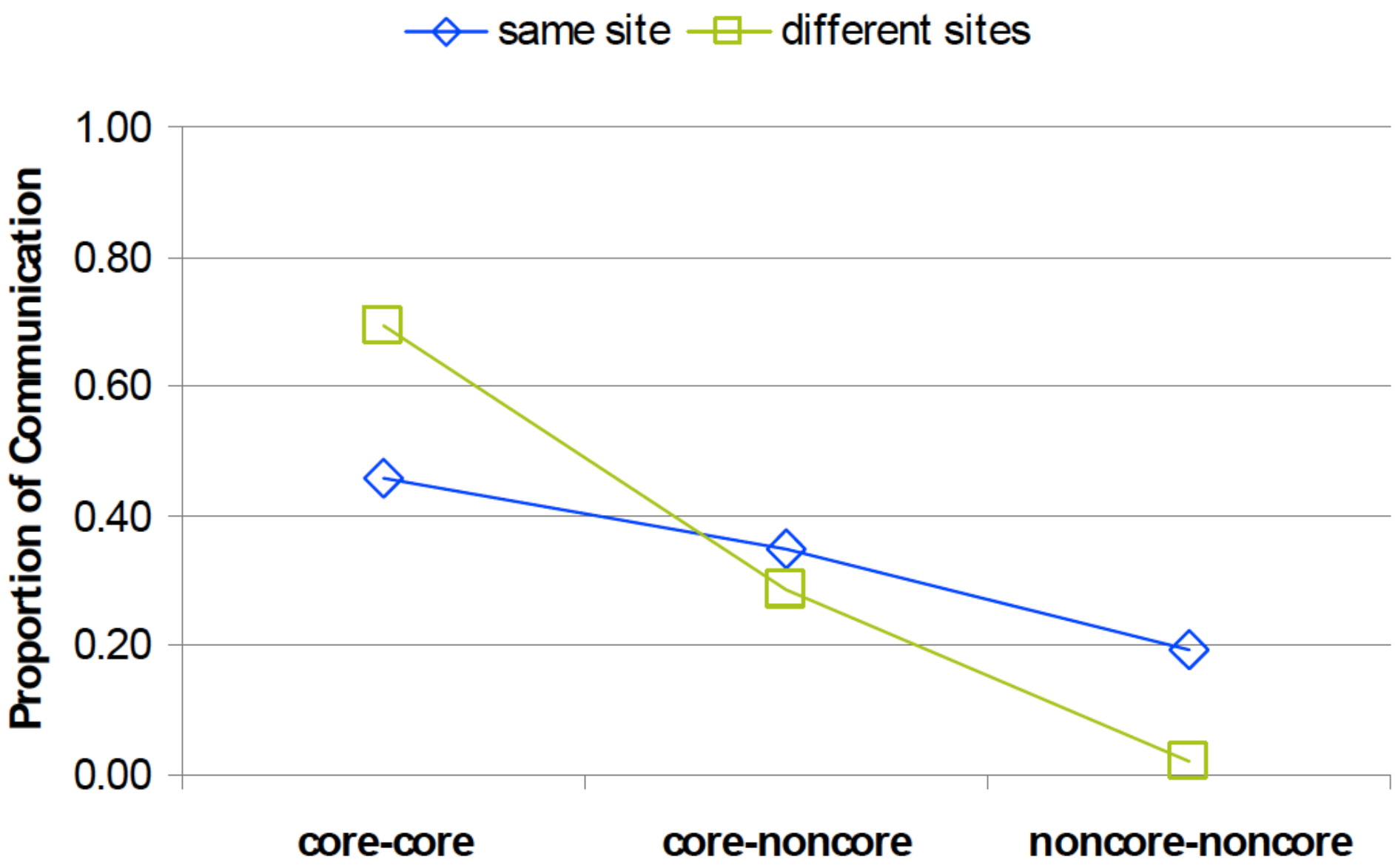
Origin of Gatekeeper findings: Allen, T. J. *Managing the Flow of Technology*. Cambridge, MA: MIT Press, 1977.

Core-Periphery Topology

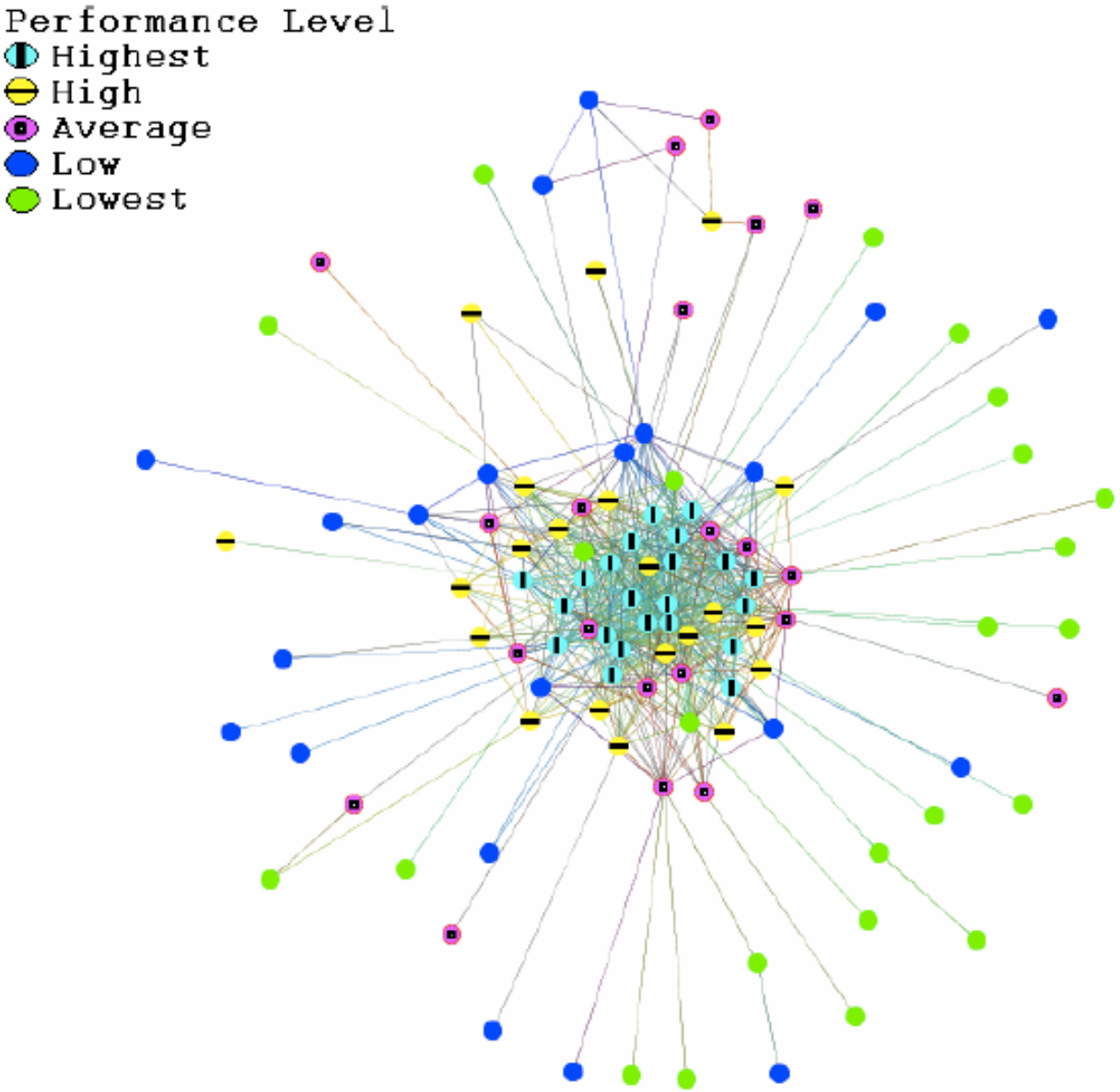


Cataldo, M. & Herbsleb, J.D. (2008). Communication networks in geographically distributed software development. In Proceedings, ACM Conference on Computer-Supported Cooperative Work, San Diego, CA, Nov. 8-12, pp. 579-588.

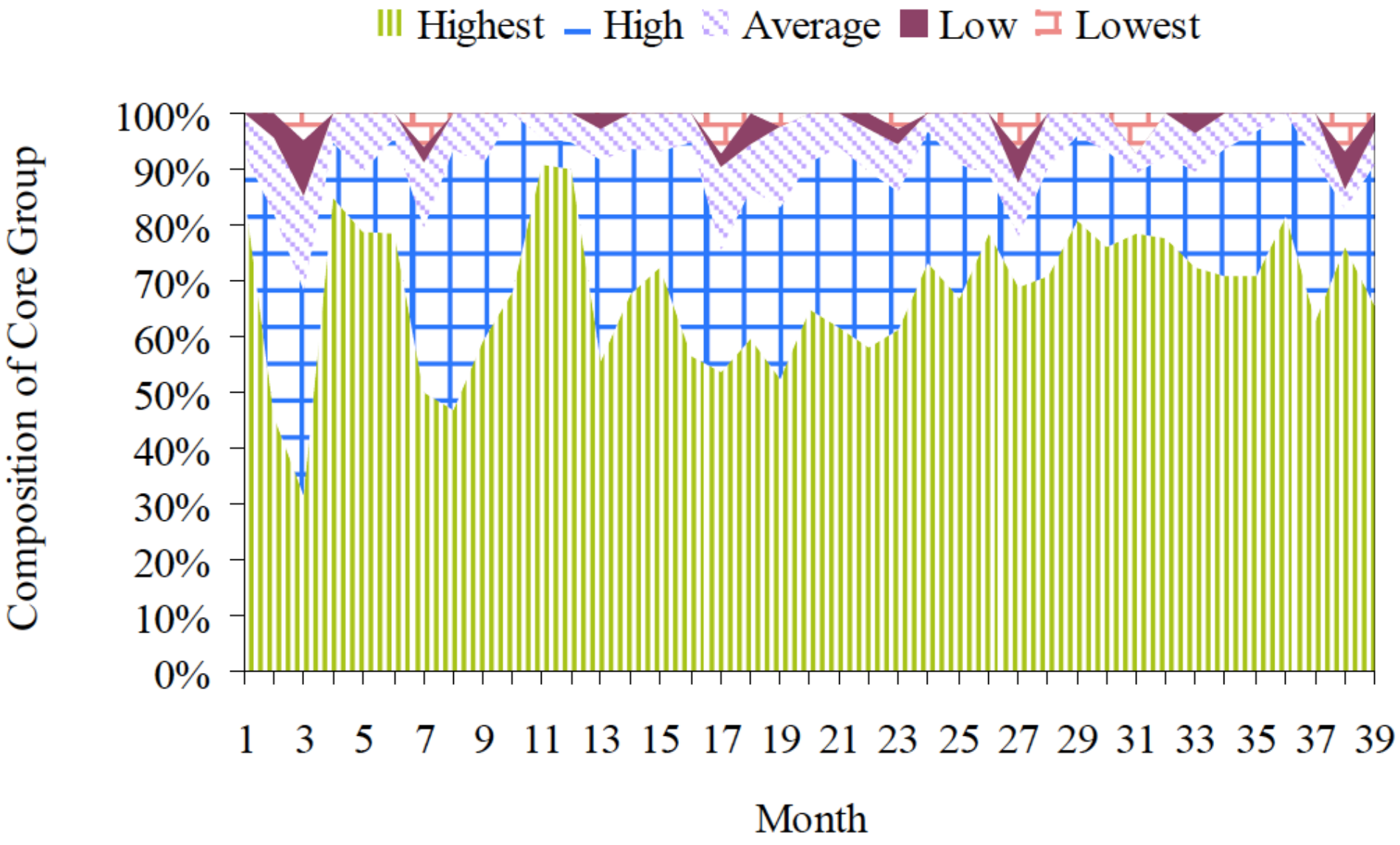
Core Handles Outside Communication



Core Membership and Productivity



Composition of Core over Time



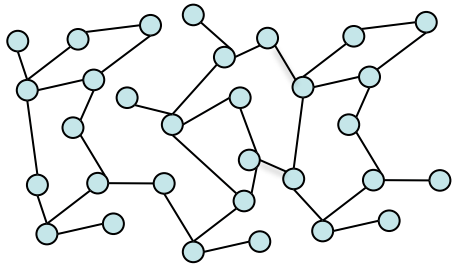
The Point . . .

- When people organize, under the right conditions they spontaneously form
  - Transactive memory systems
  - Gatekeeper networks
- Why this matters
  - Working with them provides powerful capability
  - Working against them may be necessary, but you should know what you're up against

Example: GitHub

- Why so successful?
- Provides means for humans to form and use social capabilities
  - TMS: activity traces, profiles, consistent across repositories
  - Gatekeeper networks: Watching, starring, following, curating, “asynchronous mentoring”

Socio-Technical Coordination



Decisions and Constraints

Technical coordination is a Constraint satisfaction problem (CSP) over decisions

Decisions distributed over people (DCSP)

Social algorithm to solve DCSP

Social Algorithms

- Can take advantage or fail to take advantage of powerful capabilities
- Can be derailed by people using capabilities mismatched to task
- We need a much sharper picture of these capabilities and how typical software tasks map onto them

The Science We Need

- Psychology, sociology, etc. are a starting point
- Only moderately useful in current form
  - Stretched by complexity of environment
  - Stretched by rapid change
  - Stretched by capabilities of digital tools and materials: social reach, free copying, absence of geographic boundaries
- We need a socio-technical perspective!

Next Steps?

- Collect readings
- Develop course/curriculum
- Workshops
- Special issue

A few readings I have found useful

- Argote, L. and Ren, Y. Transactive memory systems: A microfoundation of dynamic capabilities. *Journal of Management Studies*, 49, 8 (2012), 1375-1382.
- Cataldo, M. & Herbsleb, J.D. (2008). Communication networks in geographically distributed software development. In *Proceedings, ACM Conference on Computer- Supported Cooperative Work*, San Diego, CA, Nov. 8-12, pp. 579-588.
- Clark, A. From folk psychology to naive psychology. *Cognitive Science*, 11, 2 (1987), 139-154.
- Dunbar, R. The social brain hypothesis. *Brain*, 9, 10 (1998), 178-190.
- Harari, Y. N. *Sapiens: A brief history of humankind* Random House, 2014.
- Kahneman, D. *Thinking, fast and slow* Macmillan, 2011.
- Pinker, S. *The language instinct: The new science of language and mind* Penguin UK, 1995.

Q&A