Challenges and pathways to sustainability in scientific software ecosystems

http://www.flickr.com/photos/demandaj/

James Howison
School of Information
University of Texas at Austin

(based on joint work with Jim Herbsleb at CMU)

This material is based upon work supported by the US National Science Foundation under Grant Nos. SMA- 1064209 (SciSIP), OCI-0943168 (VOSS), and OAC-1453548.

How does working on things made out of digital information change the way we collectively work?

Affordance 1: Reuse

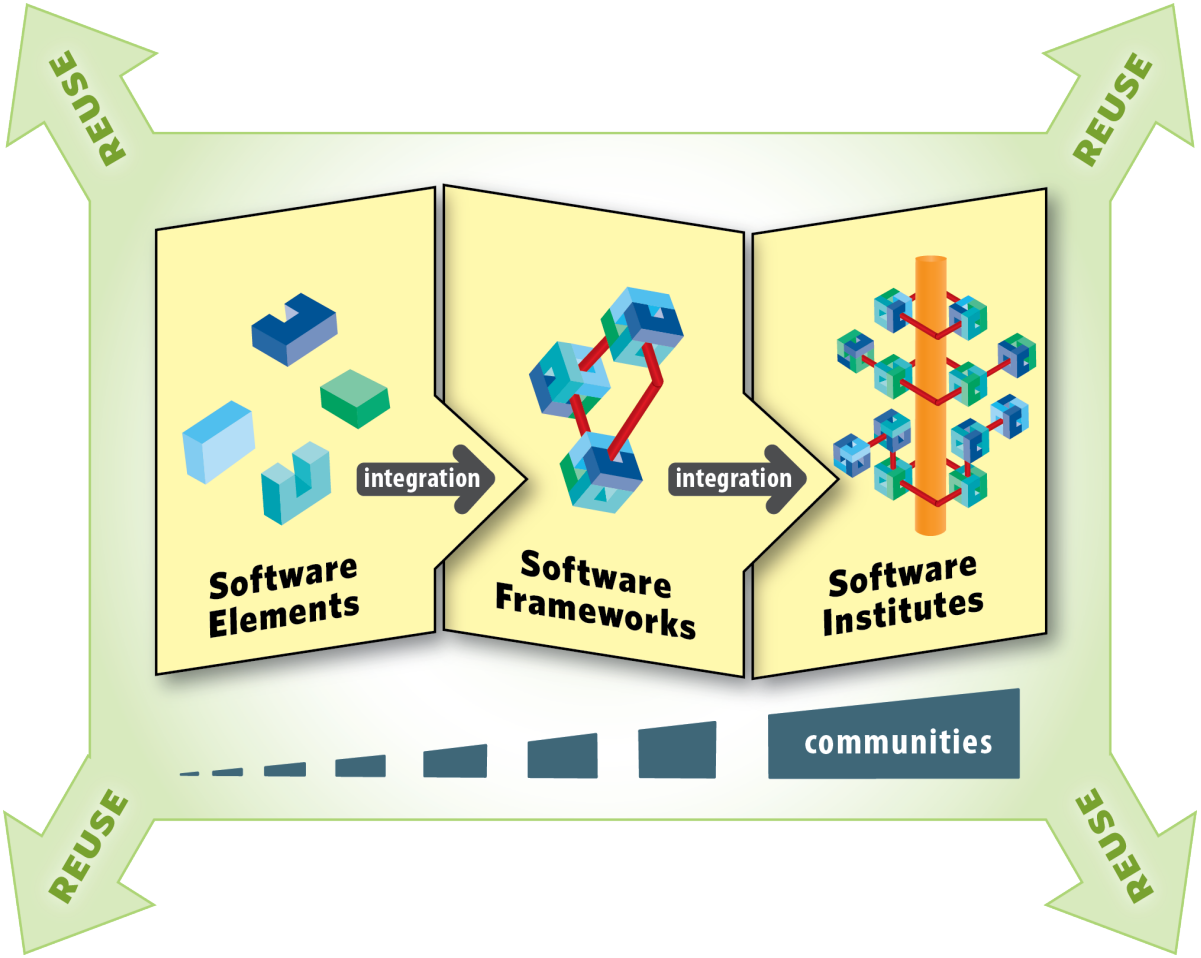
- Digital information can be copied
  - High design costs
  - Ultra-low instantiation costs
  - Cheap network distribution
- Implications:
  - “Write once, run anywhere”
  - Think of software as an artifact
  - Everyone gets a car!

Affordance 2: Recombination

- Digital information is very flexible
  - Patched
  - Wrapped
  - Extended
  - Recombined
- Re-combinability is great for innovation
  - Lots of new ways to do things

Schumpeter, Ethiraj and Levinthal, Baldwin and Clarke, Paul David

NSF's Cyberinfrastructure Vision



NSF’s “Software Vision”

“It is NSF's expectation that these investments will result in robust, reliable, usable and **sustainable** software infrastructure that is critical to the CIF21 vision and will transform science and engineering”

“These programs will result in the development of **sustainable** software communities that transcend scientific and geographical boundaries”

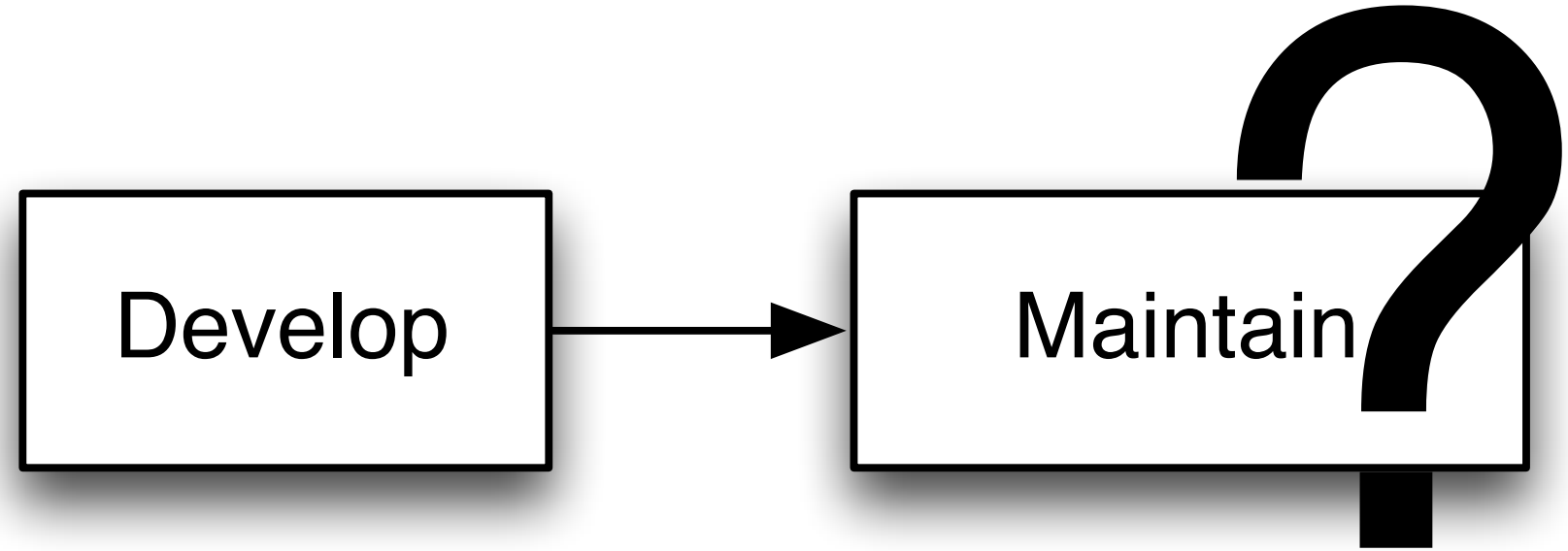
“will result in **sustainable** community software frameworks serving a diverse community”

But sustainability is elusive

I’m going to try to explain why
in a way that helps us know what to do.

https://www.flickr.com/photos/kitch/5179039214

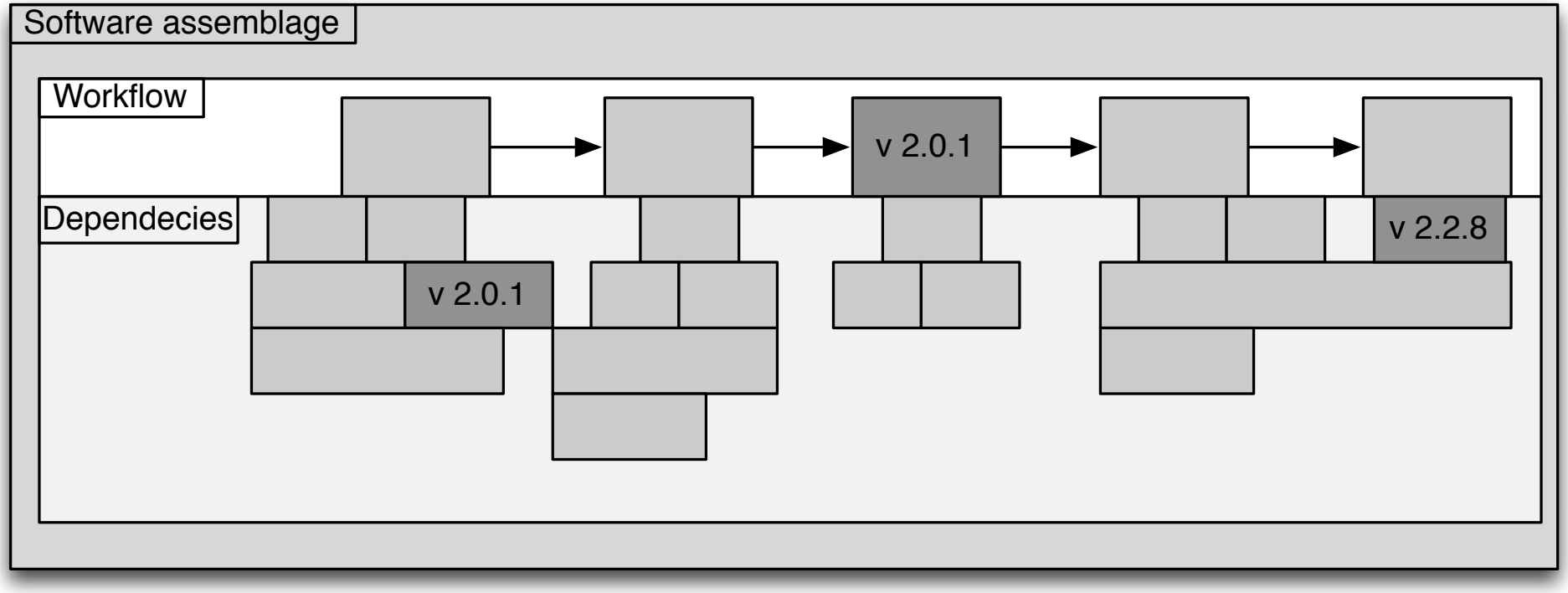
So what work needs to be done?



What drives the need for work?

1. Difficulty of production
2. Difficulty of use
3. Changing scientific frontier
4. Changing technological capabilities
5. Ecosystem complexity

How do scientists use software?



Edwards, Batcheller, Deelman, Bietz and Lee,
Segal, De Roure and Gobels, Ribes and Finholt,
Howison and Herbsleb

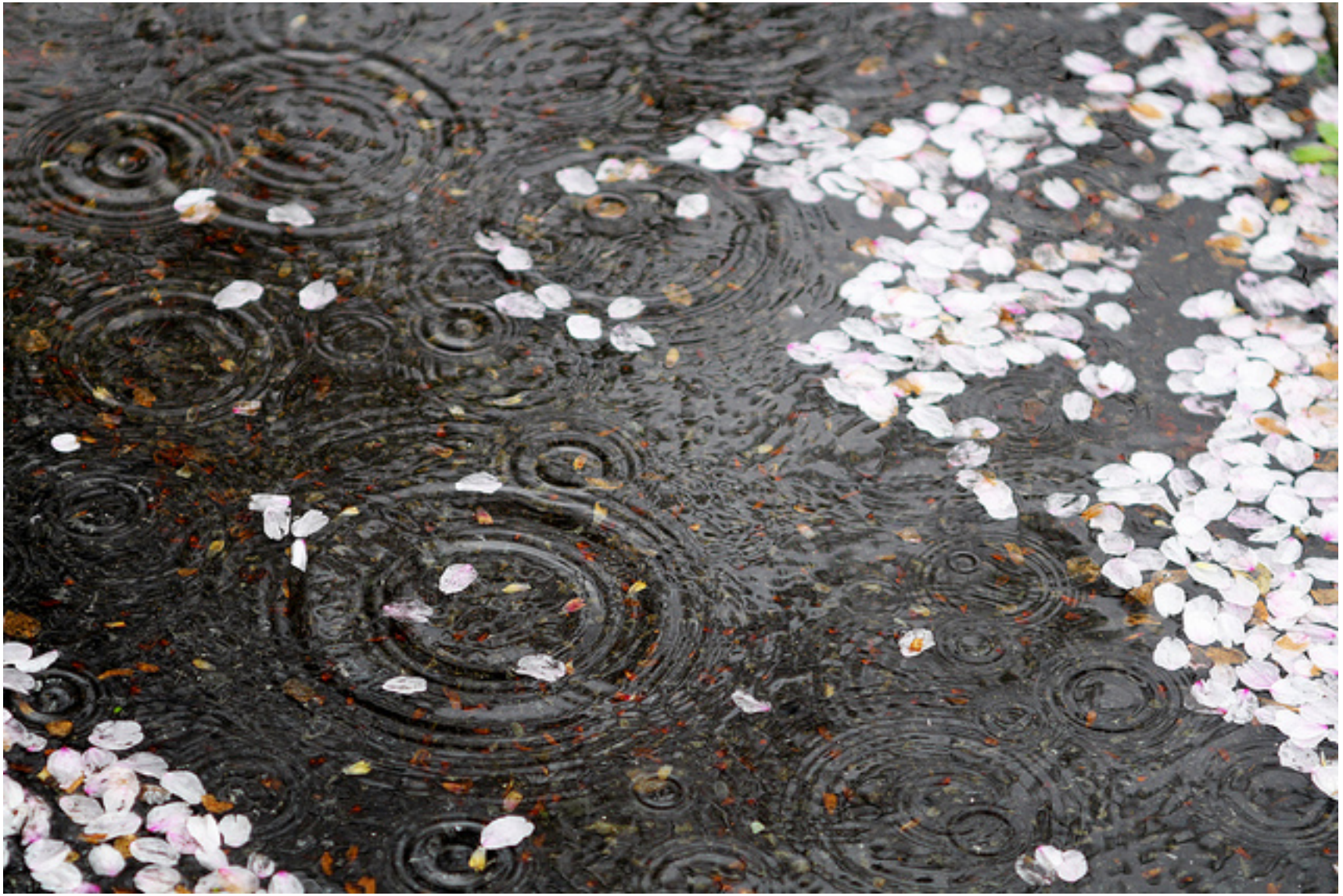
Re-animating assemblages

- Scientists pull an assemblage together, “get the plots” and often then leave it, often for months or years.
- When they return they return to *extend*; to use the software assemblage for new purposes, for new science, not simply to replicate.

But the world changes …

- Reanimation encounters change in the software ecosystem
  - Updated packages, New packages, New interfaces
- And not just in the immediate components of a workflow, but in the dependencies.

- This work is echoed at component producers, since components are themselves assemblages.

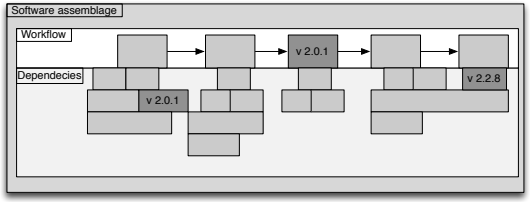


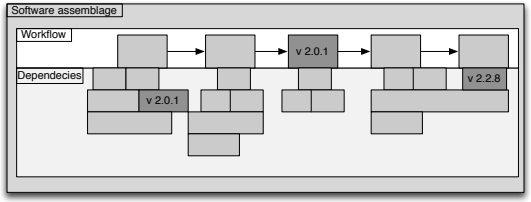
https://www.flickr.com/photos/mrhayata/6933963596

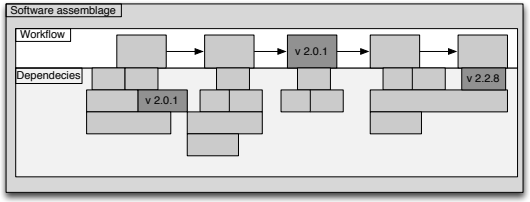
What holds a complex software ecosystem together (if anything)?

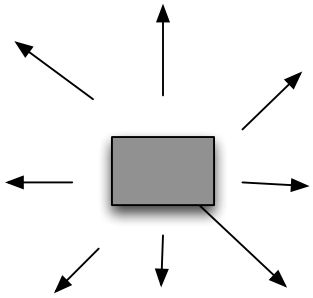
- Sensing work
  - knowing how things “out there” are changing
- Adjustment
  - making appropriate changes to account for changing surroundings
- Synchronization
  - ensuring that changes in multiple components make sense together, avoiding cascades.

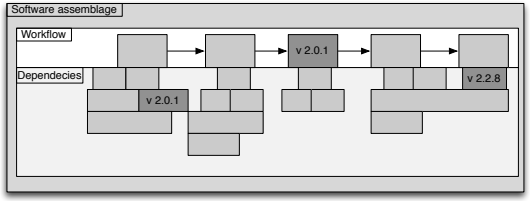
Number of users (reuse)

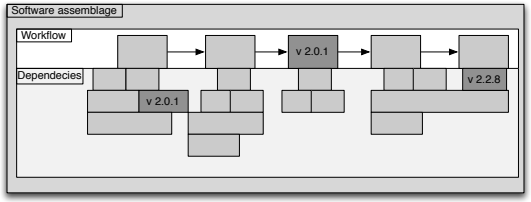


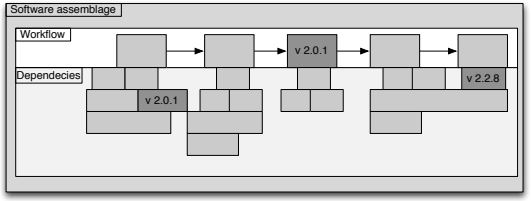




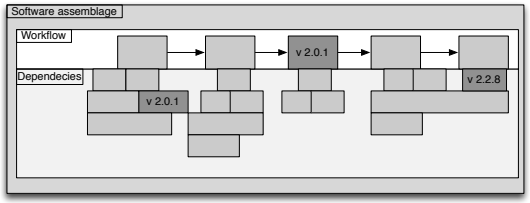


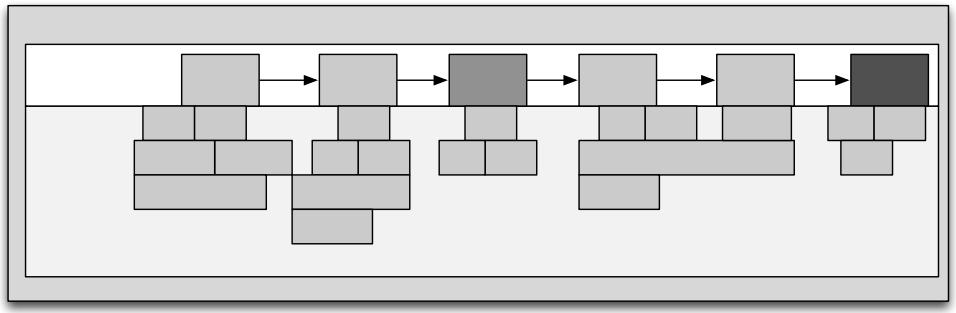


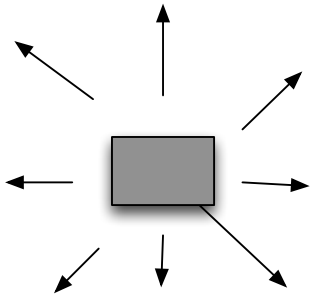


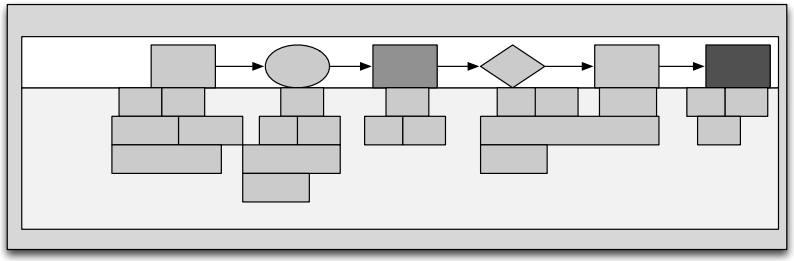


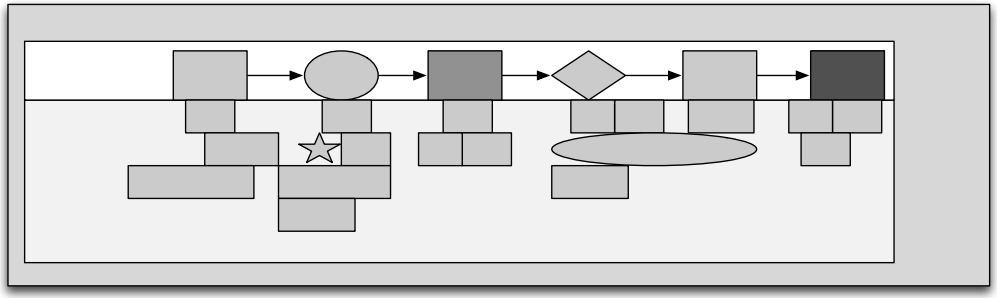
Diversity of use (recombination)



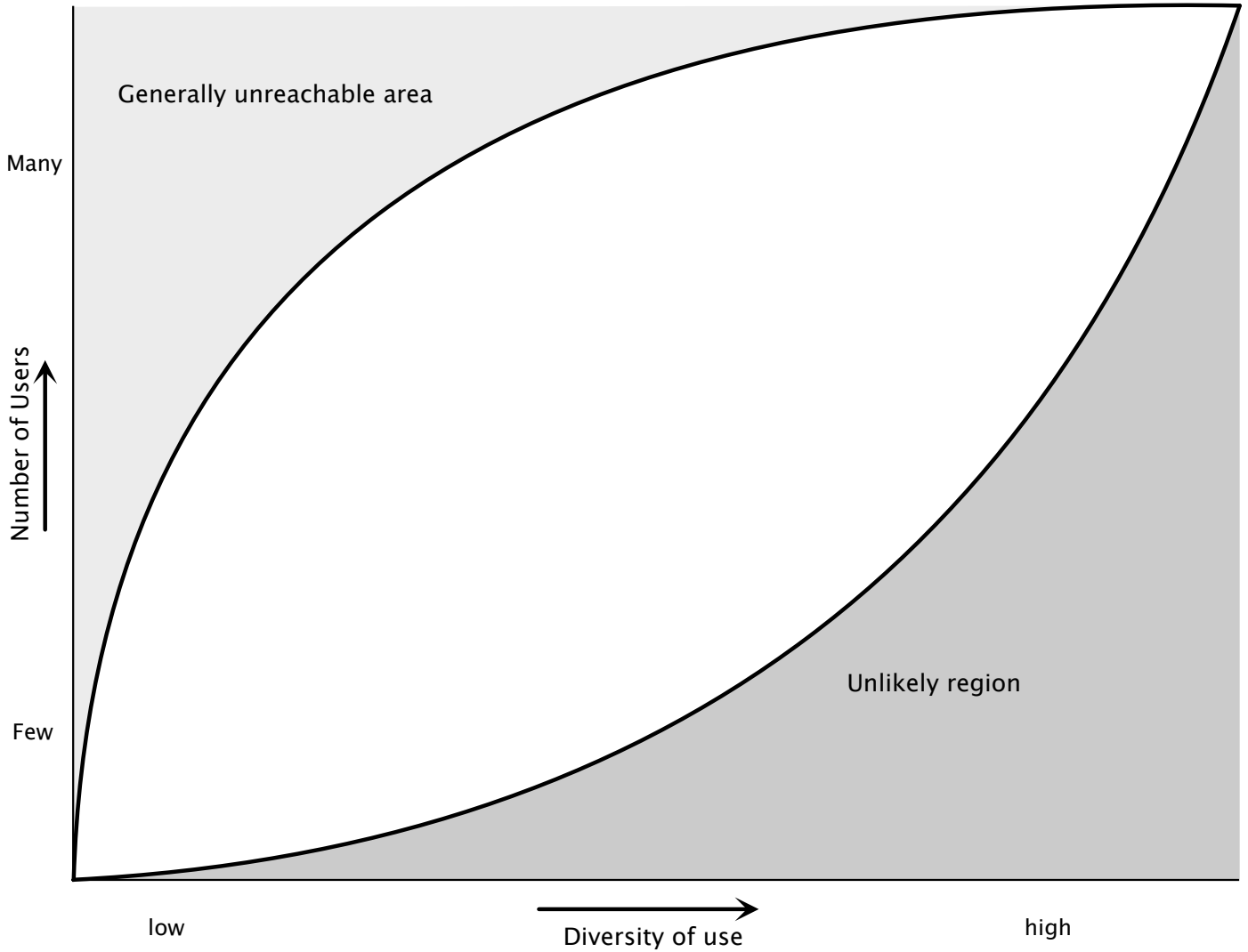








Ecosystem Position

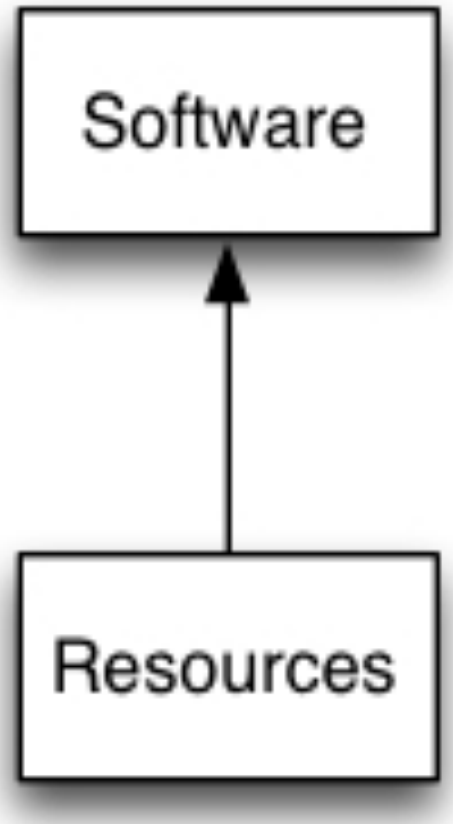


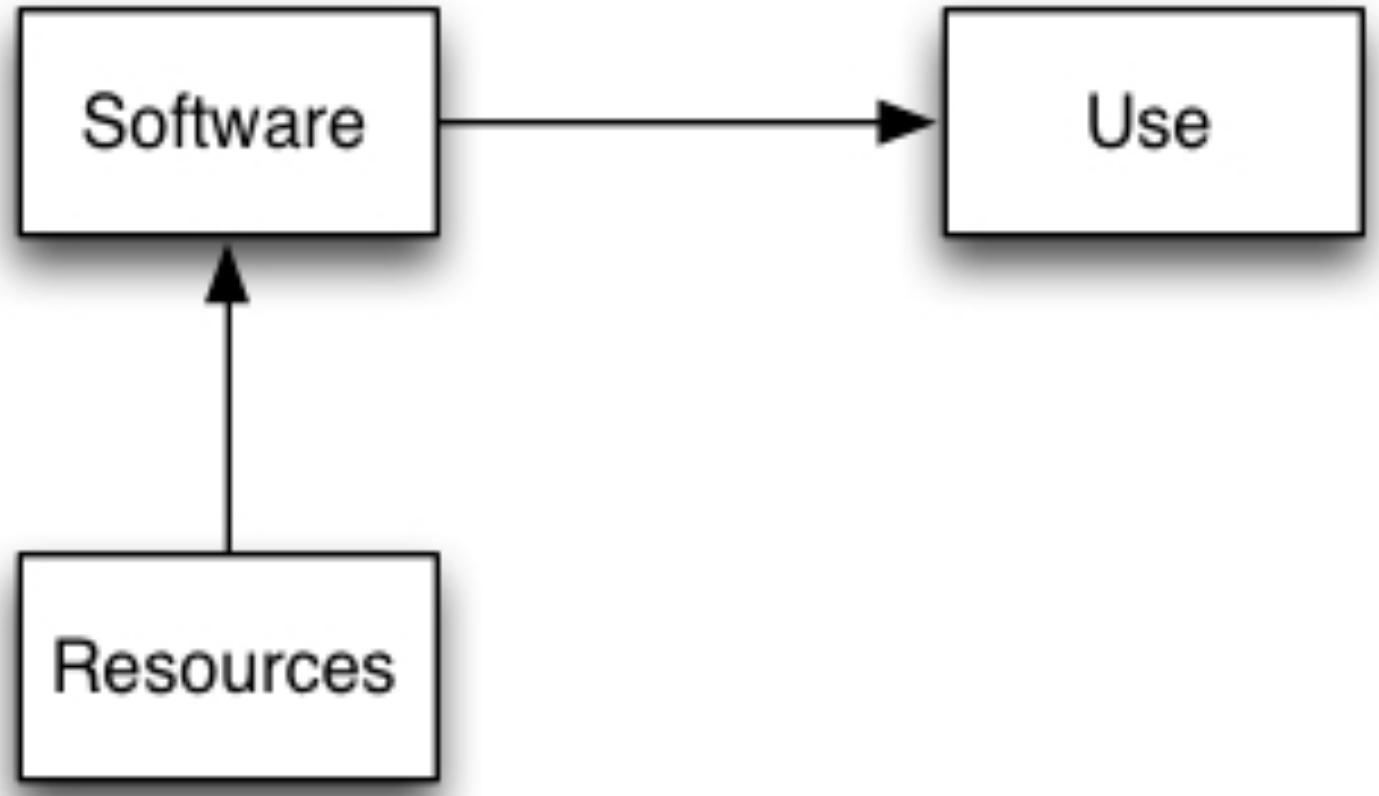
How do different kinds of needed work scale with ecosystem position?

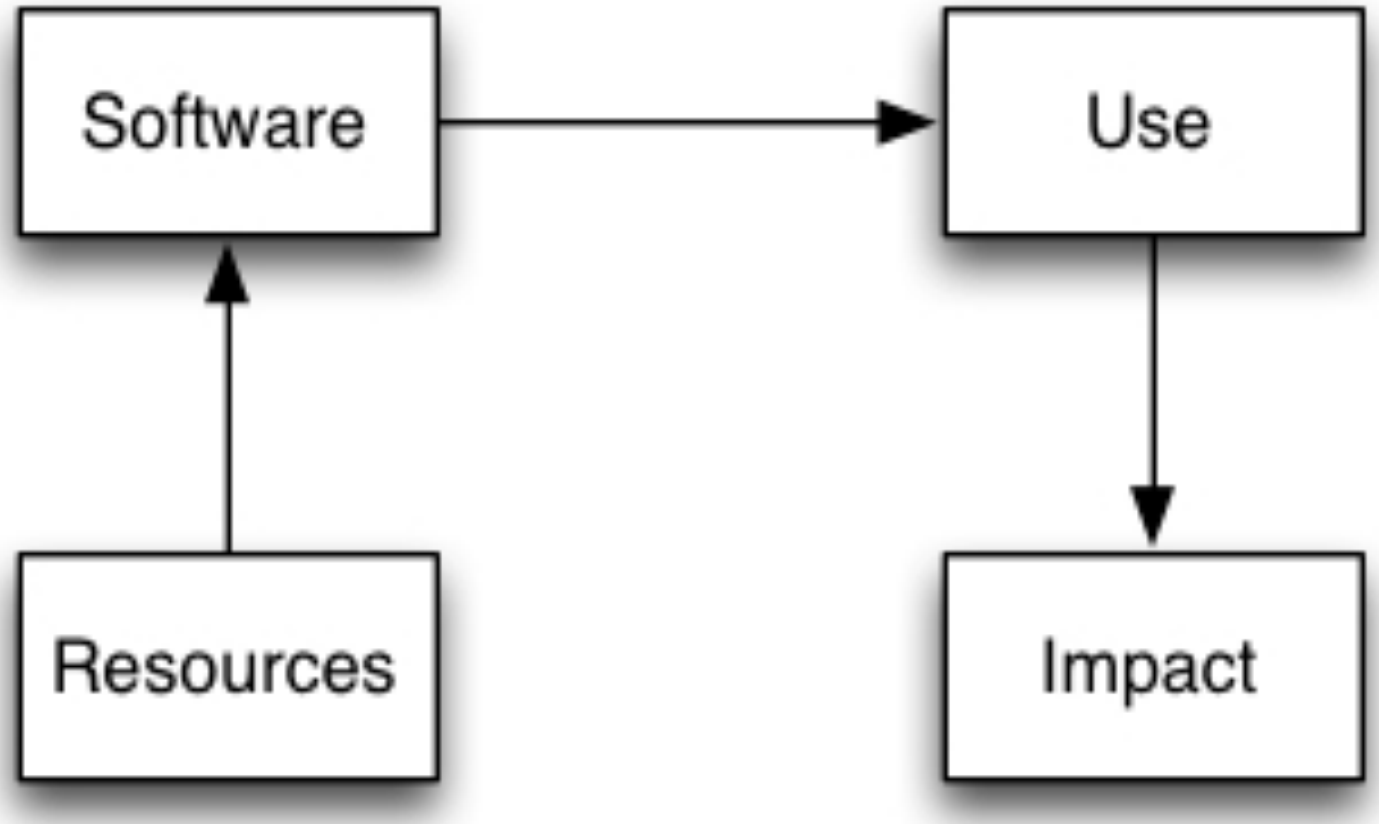
- New feature development/Technology adaptation
  - At most linearly across both dimensions
- User support
  - Linearly, perhaps even reducing at high numbers as users support each other
- Sensing work
  - Linearly with diversity of use
- Adjustment and synchronization
  - **Exponentially** with diversity of use (recombination)
  - Even assuming a constant rate of change of complements

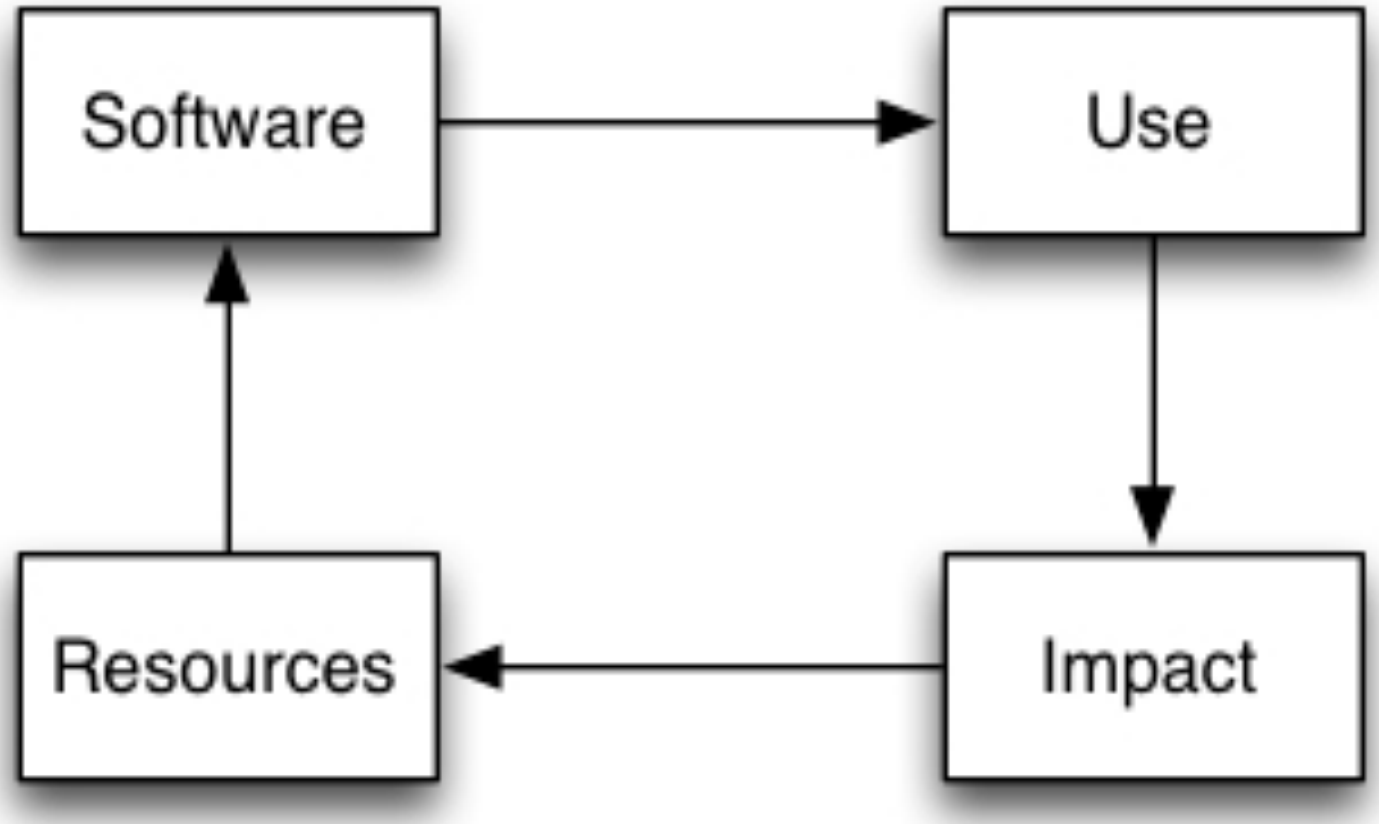
Holding things together is hard work

But you can’t unlock the potential of cyberinfrastructure without it


Software
Resources


Software
Use
Resources


Software
Use
Resources
Impact

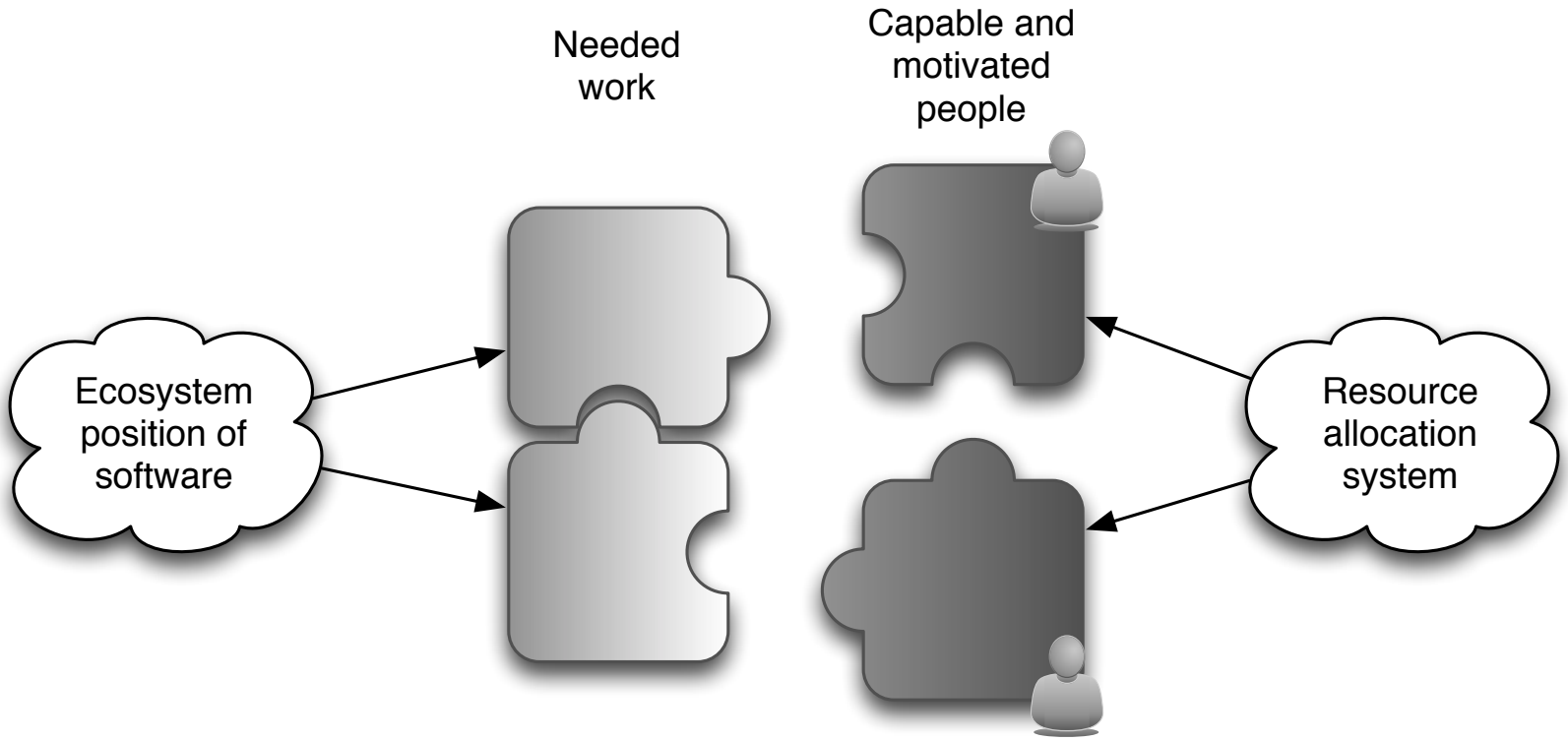

Software
Use
Resources
Impact

Questions of sustainability

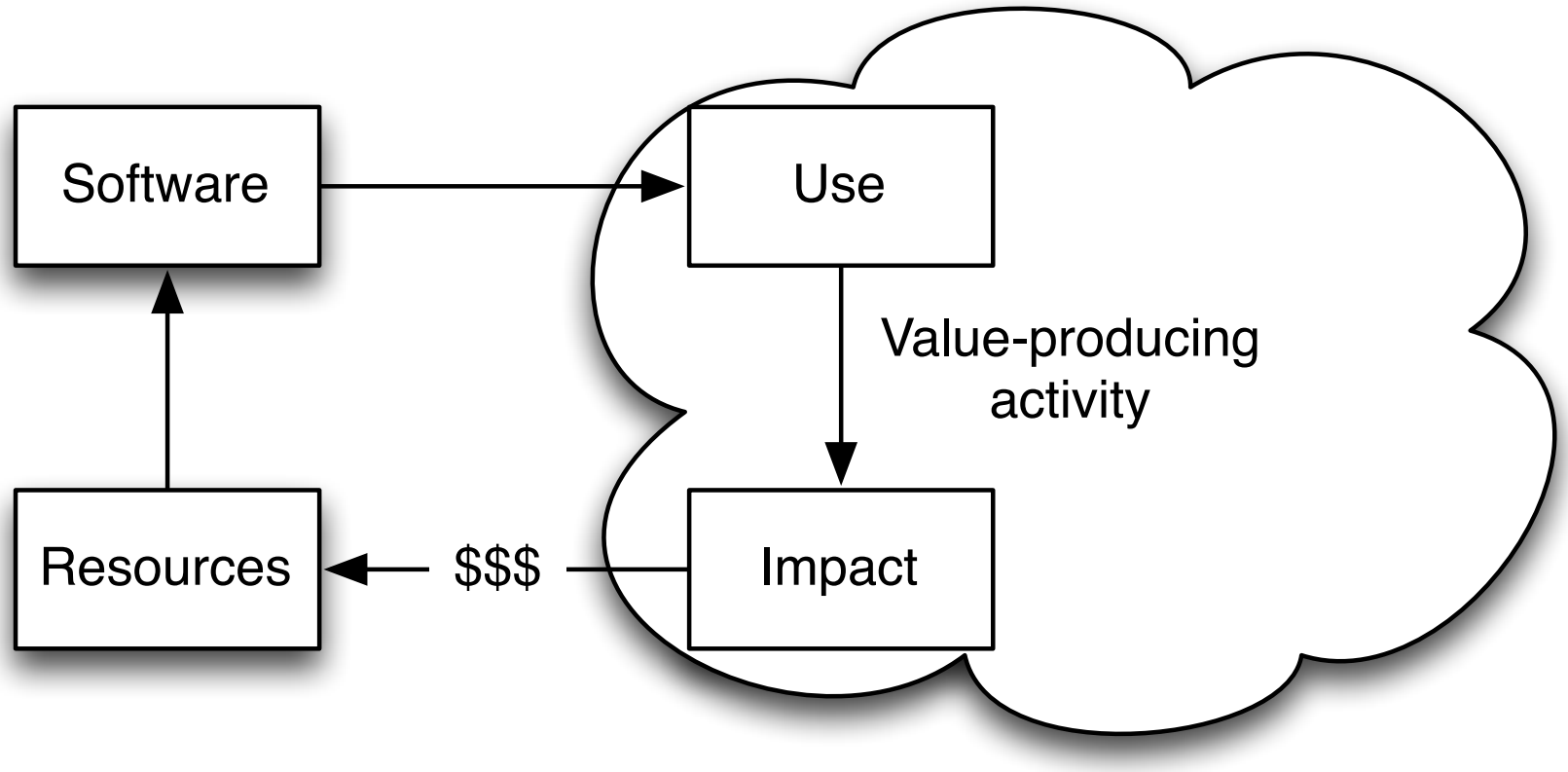
- How, and to what extent, does a project attract new resources?
  - Turn use and impact into more resources?

- How does a resource attraction system handle sensing, adjustment, and synchronization work?

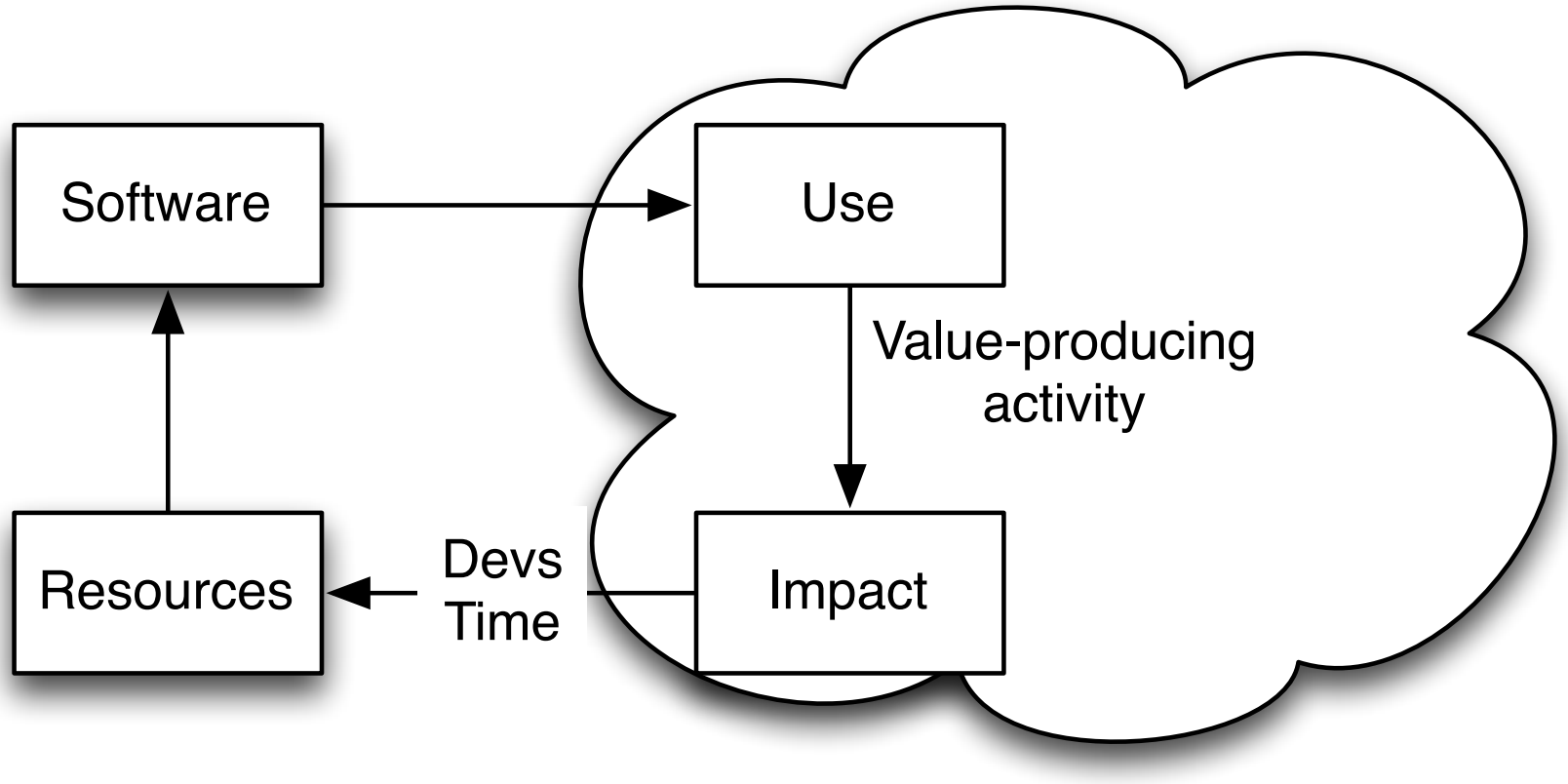
Bringing it all together



A commercial project

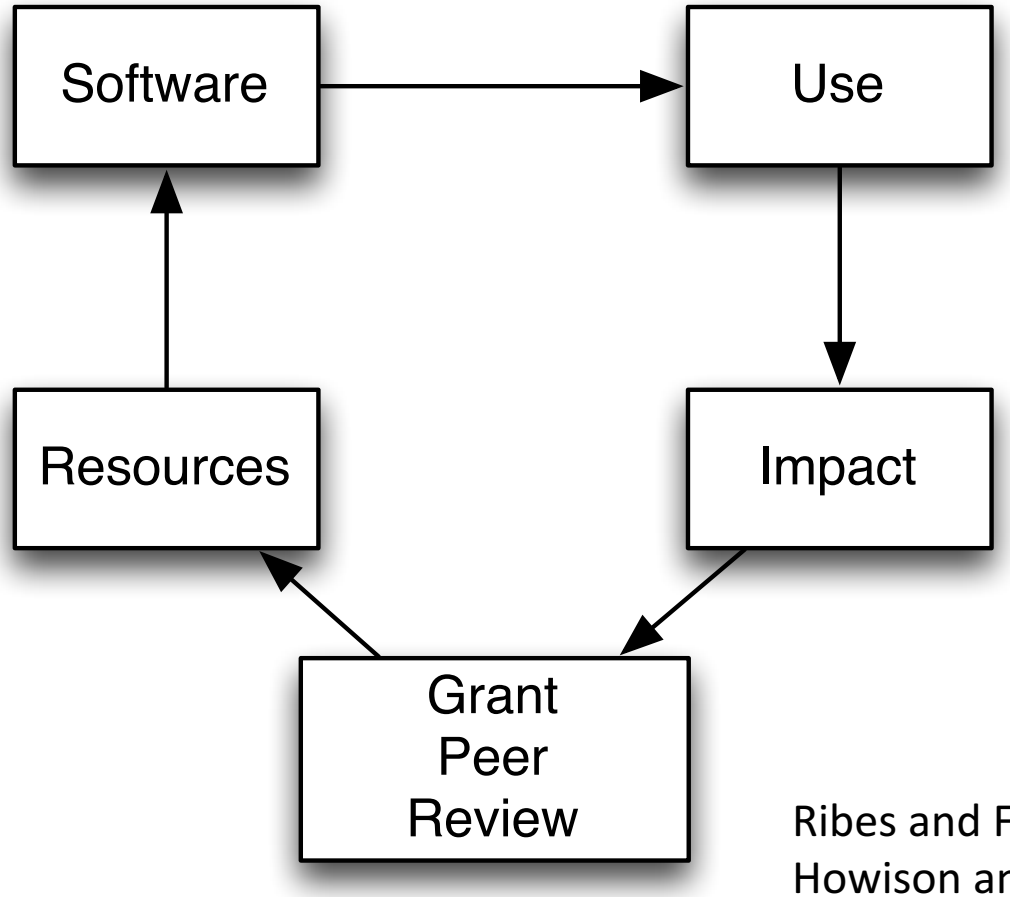


Open Source Peer Production



Howison and Crowston (2014) Collaboration through Open Superposition. *MIS Quarterly*

Grant-making

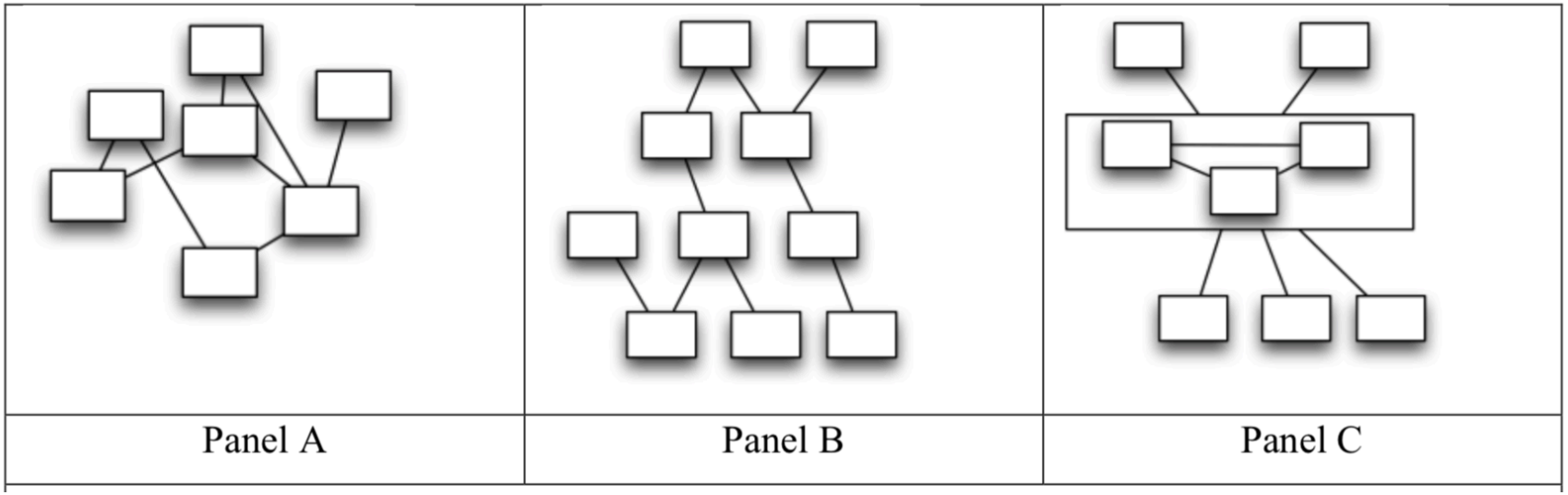


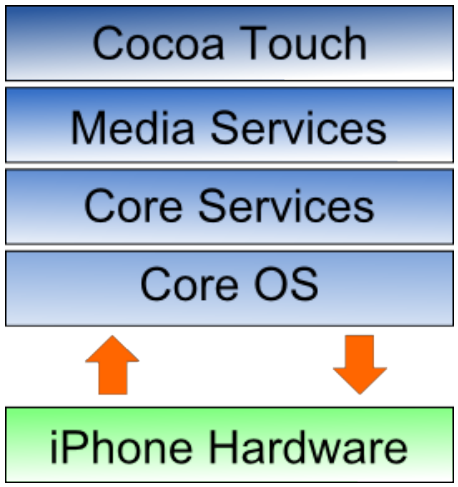
Ribes and Finholt 2007;
Howison and Herbsleb 2011

How do resource attraction systems cope with challenges of ecosystem complexity?

Markets and ecosystem complexity

- Sales provide insight (sensing) at the same time as they generate resources
  - Fund some adjustment work internally
- Emergence of Platform strategy:
  - Suppress complexity by disallowing recombination between customer apps.
  - Sensing undertaken by code review (via app store submission)
  - Adjustment and synchronization through new releases of the API





The platform strategy **requires power** and a willingness to use it to *reduce* recombination complexity.

"Anyone who doesn't do this will be fired. Thank you; have a nice day!"
(Yegge Memo about Amazon)

Photo credit: http://www.flickr.com/photos/smemon/5324223435

Attempting command and control

- Scientific software developers sometimes appeal for hierarchical control
  - "The funder should just make it mandatory"
  - "We should meet and plan a roadmap"
  - All focused on suppressing complexity
- But digital flexibility inhibits this
  - Scientists are going to tinker, new technologies are going to suggest new technologies

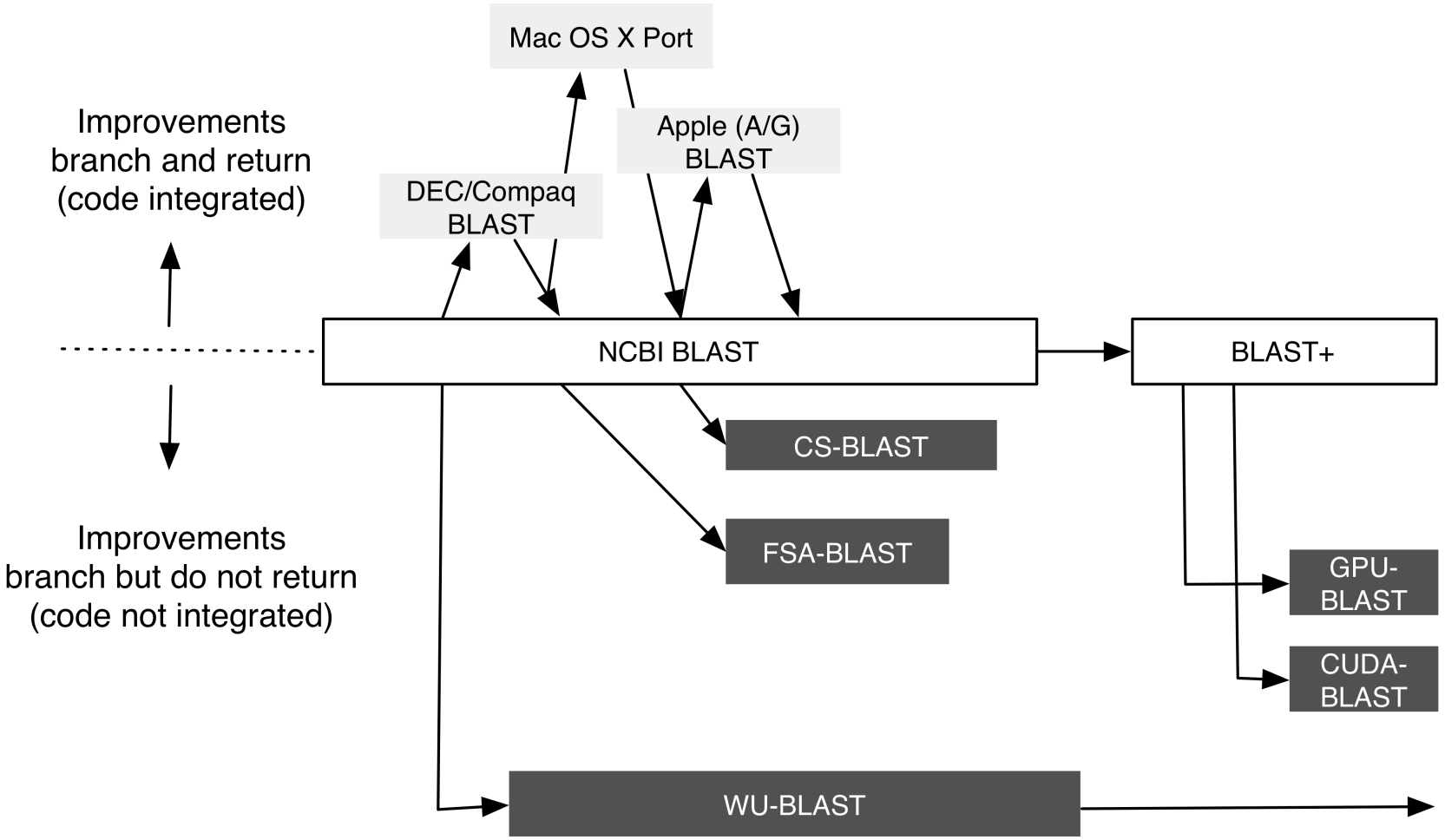
Open source peer production

- Sensing:
  - Contributions from the edge, driven by use value, provide **direct insight into usage**.
- Adjustment:
  - **Pushing upstream** uses the cheap copies affordance of software to scale adjustment.
- Synchronization:
  - Emergence of **distributions** which collate the adjustments and attempt synchronization
    - Debian, Red Hat, Eclipse

Grant funding

- Service center argument reduces input from the edge, places burden on core team
- Little visibility into use context of software
  - Developers are not front-line scientists
  - No scalable way to see what is happening in use contexts
- Success via focusing on low diversity of use/high user numbers
  - Works in specific locations (e.g., BLAST)
  - Often impossible, so focus on “power users” (Batcheller and Edwards; Bietz and Lee)
- Inherent tensions undermine sustainability

Academic reputation makes this worse



James Howison and Jim Herbsleb (2013) *incentives and integration in scientific software production*. Proceedings of ACM conference on Computer Support Cooperative Work.

Grant making and sustainability

Grant-making has limited built-in feedback that facilitates sensing, adjustment and synchronization work.

Cyberinfrastructure discourse invokes both reuse and recombination affordance

But its resource attraction system design is founded only on the reuse affordance

What is to be done?

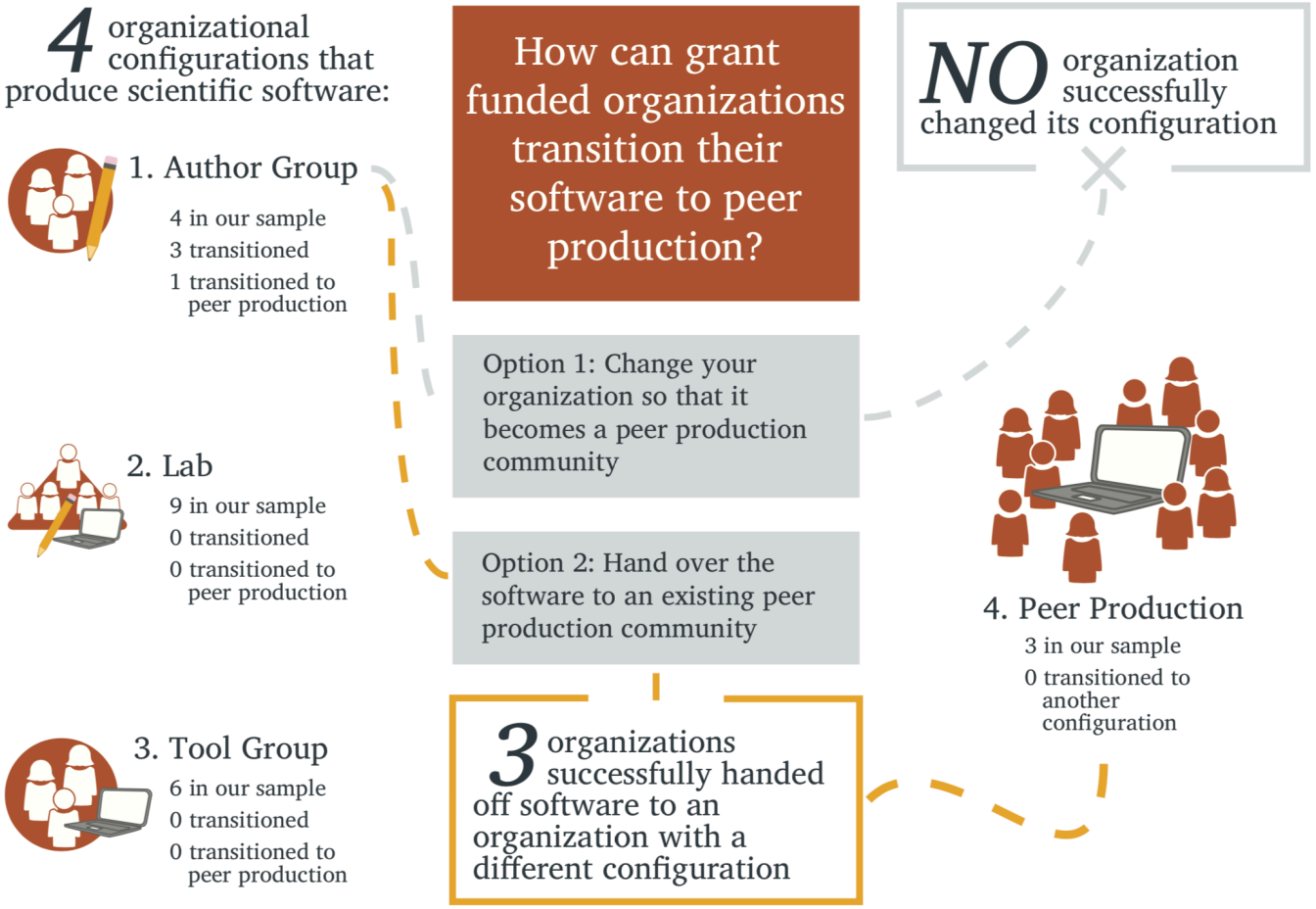
Policy responses

- Policy Response 1:
  - Transition the resource attraction system
    - From grants to commercial
    - From grants to open source peer production
- Policy Response 2:
  - Change grant-making to incentivize sensing, adjustment and synchronization work

Project to Commercial

- Assuming a sufficient market size!
- Change from free provision to paid provision
  - Now must manage business
  - Users must have budget
  - Appears that funding agency is “paying many times over” (thinking artifact not work)
  - Large customers (e.g., Wall Street) can pull attention

Project to Open Community?



Johanna Cohoon, & James Howison. (2018). Routes to Sustainable Software in Science: Transitioning to Peer Production. Presented at the Academy of Management Conference, Chicago, IL.

Howison, J. (2015). Sustaining scientific infrastructures: transitioning from grants to peer production (work-in-progress). Presented at the iConference, Irvine, CA. Retrieved from https://www.ideals.illinois.edu/handle/2142/73439

Response 2: Make different grants

Fund, but more importantly develop reputational rewards to, innovation in:

1. Sensing
2. Adjustment
3. Synchronization

Improving Sensing

Fund, but more importantly develop reputational rewards to, innovation in sensing the scientific software ecosystem:

- Measure diversity of use contexts, understand how users recombine: Know what *generates* your work!
  - Go beyond single tool user-studies. How do they *combine* your tool?
- Software that reports its own use?
  - Overcome concerns about visibility and scientific competition
  - Yes, privacy matters, but users have responsibilities as well.
- Increase visibility of software in publications
  - See Citation Workshop on Wednesday.
  - CiteAs.org

CiteAs [1]
alpha

Improving Adjustment

- Incentivize projects to be open to gathering and rationalize outside adjustments.
- Overcome the “service center” framing
- Inculcate stewardship orientation within grant funded projects.
- Projects must not just “be open” but contributing upstream and downstream.

Improving Synchronization

- Fund software distribution work and innovation in distribution
  - Distributions can manage cascades
- Opportunities for research including simulating ecosystem impact of changes
  - If we knew how tools, data and questions were linked we could test possible changes.

When we organize work on digital information

- The affordance of cheap copies matters
  - but it isn't the whole story
- Digital flexibility leads to recombination and ecosystem-level complexity
  - This super-charges the sustainability problem
- Resource attraction system must address complexity
  - Markets and peer production have feedback from "the edge"
  - Grant-based want widespread recombination but has not yet grappled well with complexity

Change the conversation

- Don't ask what someone has or wants to build, ask what they've recently pushed upstream
- Ask how does my project affect the distribution of work in the ecosystem? Do we understand what work our changes are making for our users? Could the work we are doing be better done elsewhere?