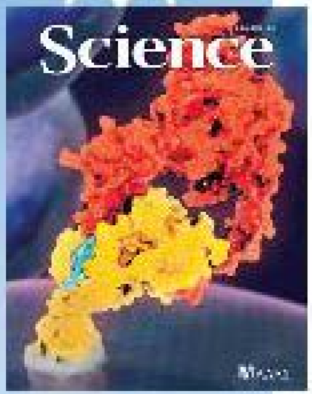
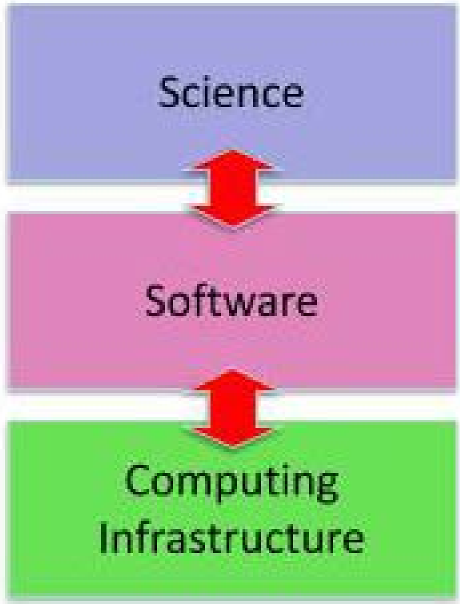
Software





- Software (including services) essential for the bulk of science
  - About half the papers in recent issues of Science were software-intensive
  - Research becoming dependent upon advances in software
  - Wide range of software types: system, applications, modeling, gateways, analysis, algorithms, middleware, libraries
  - Significant software-intensive projects across NSF: e.g. NEON, OOI, NEES, NCN, iPlant, etc
- Software is not a one-time effort, it must be sustained
  - Development, production, and **maintenance** are people intensive
  - Software life-times are long vs hardware
  - Software has under-appreciated value

Challenges - Career Paths

- People are essential elements of research infrastructure - they need:
  - Education and training to be productive
  - Career paths to remain motivated
  - Incentives to move along their career paths
- It's difficult to motivate researchers to create sustainable software - why?
  - Few research career paths available for supporting software
  - No incentives for researchers to develop broad skill sets outside of domains
  - Substantial competition from private companies
- Is there a role (career path) for non-tenure-track researchers who produce software, data, etc. in universities?
  - Assuming yes, do universities recognize and support this?
    - If no, how to get them to?

Challenges – Skills Retention and Training

- Significant student and “early stage researcher” labor
- Prevalence of idiosyncratic architectures needing out-of-the-mainstream skills
- Turnover (students graduate, staff are hired away)
- Software development best practices (e.g. Agile) not well understood or not easily transferable to the scientific environment
- Q: What software engineering practices work in science software?
  - Barry Boehm: Balancing Agility and Discipline

Challenges – Scientific Software is Inherently Interdisciplinary Work

- Scientific software contributors work in both computer science and another science or engineering area, or even multiple areas
- Other fields require significant immersion to understand and contribute
- Doesn't fit the academic research silos
- Is often discouraged.

Challenges – Evolution

- Portability: How to deal with changing hardware, middleware, and languages?
- Multiple dominant architectures: Do Cloud vs. HPC architectures and software stacks need to converge?
- Scaling without help from Moore's Law: Useful software needs to scale as more users adopt it for larger problems

Challenges - Dissemination

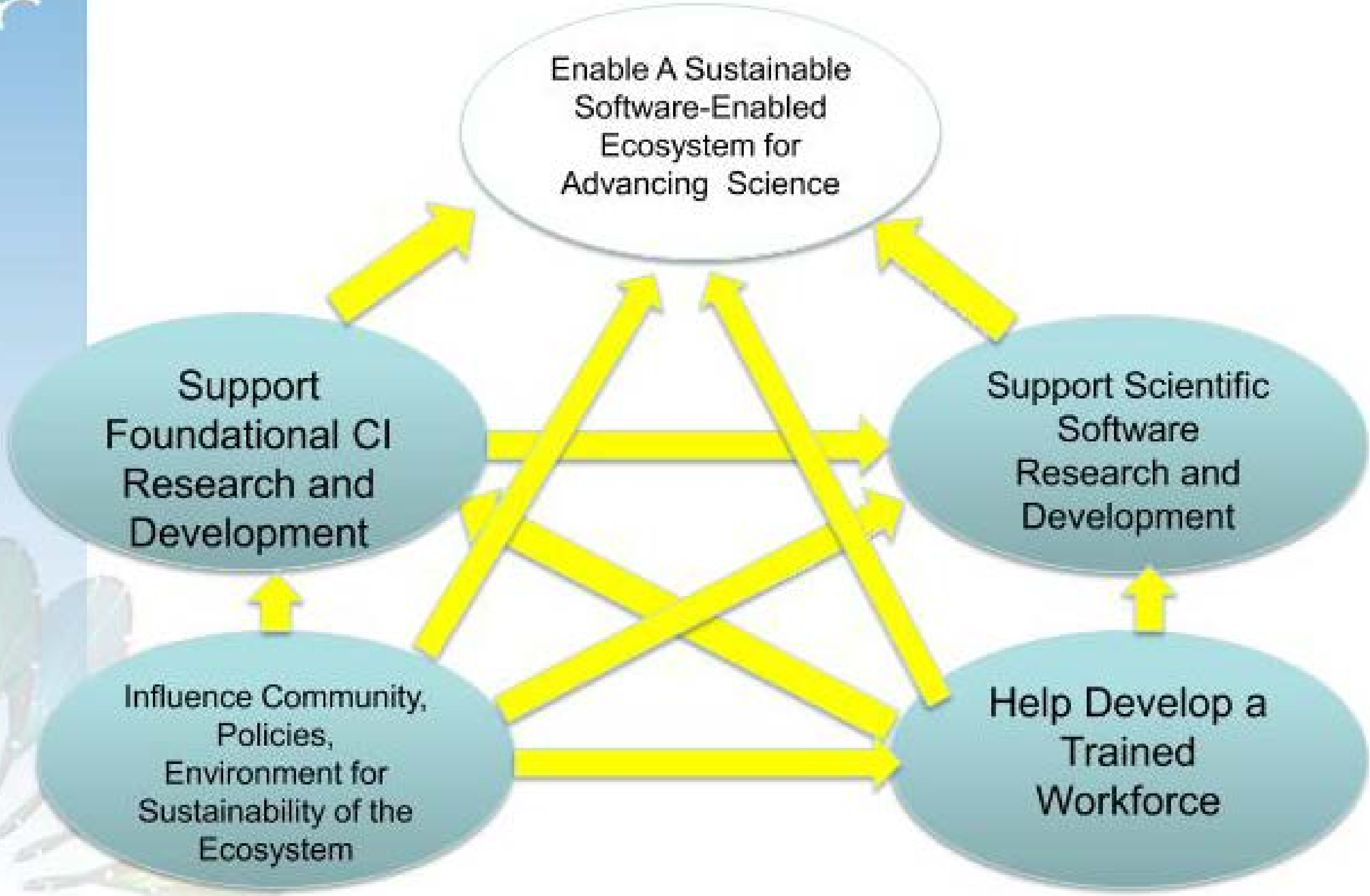
- Making software findable
  - EAGER: Semantic software discovery
- Documenting the available software
- Providing examples of use
- Characterizing strengths, weaknesses, boundaries of application
- Sharing experience of other users.

DISCUSSION

Research Software vs Infrastructure Software

- Some software is intended for research
  - Funded by many parts of NSF, sometimes explicitly, often implicitly
  - Intended for use by developer
- Other software is intended as infrastructure
  - Funded by many parts of NSF, often ACI, almost always explicitly
  - Intended for use by community

ACI Software Cluster Strategy

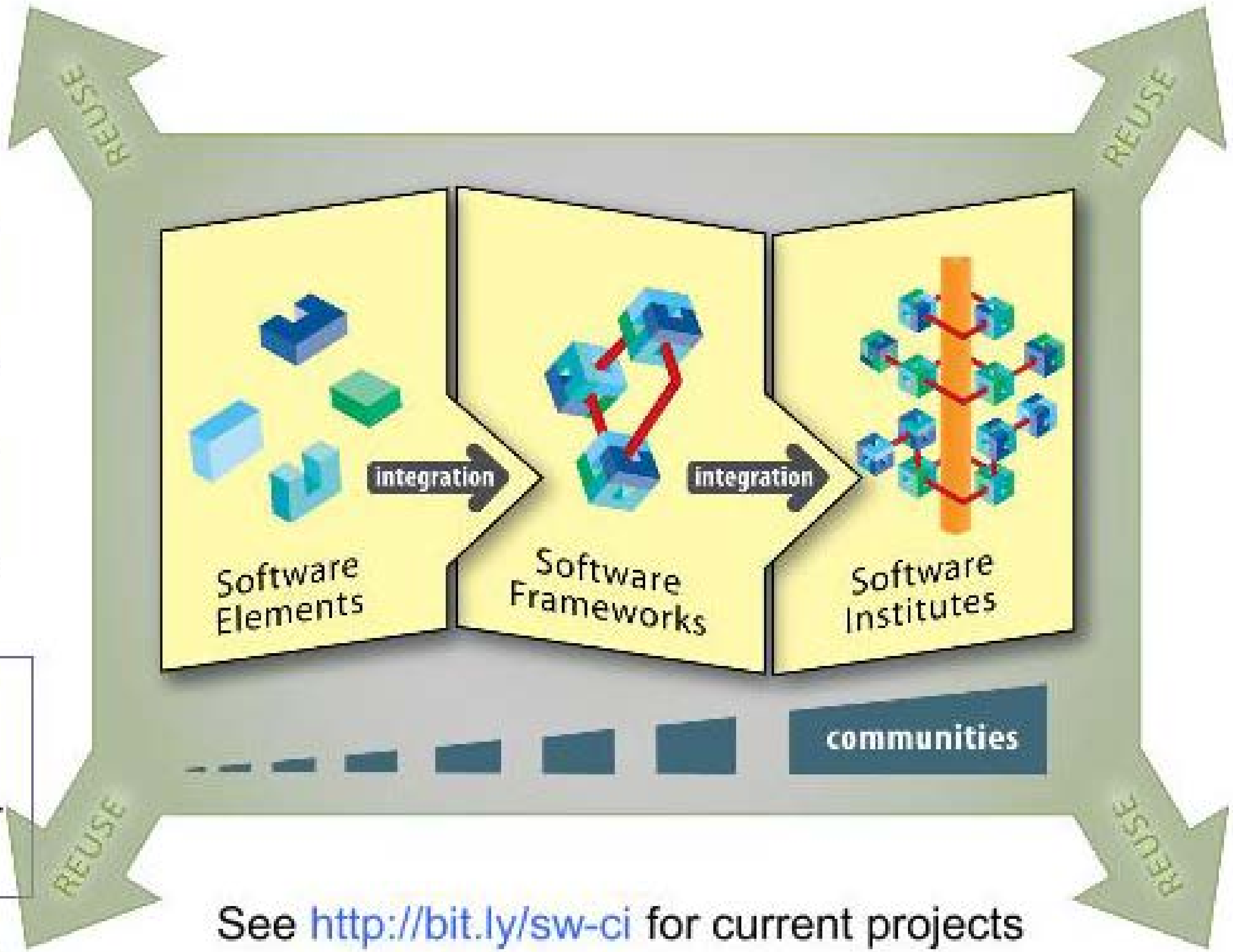


NSF Software Infrastructure Projects &
Software Infrastructure for Sustained Innovation (SI2)

SI2: 5 rounds of funding, 65 SSEs

SI2: 4 rounds of funding, 35 SSIs

SI2: 2 rounds of funding, 14 S2I2 conceptualizations. 1-2 implementations



See http://bit.ly/sw-ci for current projects

Also: EAGERS, RAPIDs and Workshops to target areas of opprtunity

Program Priorities

- Multidisciplinary and omni-disciplinary software as a national software cyberinfrastructure
- Software that builds on other ongoing NSF-supported programs.
- Techniques, tools and processes for rapid integration of software that reduces cost of custom solutions and custom integrations
- Embedded innovation and research on the development, effectiveness, usability, adoption, and organizational aspects of the software and the project.
- Serious considerations of security, trustworthiness and reproducibility.
- Comprehensive, innovative approaches to sustainability (e.g. SAAS, incorporation into university offerings, commercialization)
- Science-inspired education and LWD
- Comprehensive metrics (ideally impact)

Sustainability - Motivation

- Arfon Smith (GitHub) keynote: Scientific Software and the Open Collaborative Web
  - Example from data reduction in astronomy, where he needed to remove interfering effects from the device; work needed was persistent, but there was no practice of sharing this, so many researchers repeated the same calculations; ~13 person-years were wasted
- Why don't we do better?
  - Because we are taught to focus on immediate research outcomes and not on continuously improving and building on tools for research
- When we do know better, why we do not act any different?
  - Due to incentives and their lack: only the immediate products of research, not the software, are valued
- Open source community has excellent cultures of code reuse, where there is effectively low-friction collaboration through the use of repositories
  - This has generally not happened in scientific software

Challenges

- **Sustained National and International Funding Models**.
- **Career paths for software-focused researchers**. University structure and academic culture rewards publications; what about researchers whose main products are software?
- **Incentives, including credit**. How should software be cited? How are software contributions recognized?
- **Skills Retention and Training**. What software engineering practices work in science software?
- **Inherently Interdisciplinary work**. Cross-over knowledge development, credit
- **Evolution**. Technology evolution. Expanding needs.
- **Dissemination.** Making software available and experiences widely known

Challenges - Funding Models

- University funding model:
  - Large number of universities
    - Public (state-funded, not federally-funded), private, for profit
    - No direct national funding
  - Indirect funding of education through students
  - Indirect funding of research through projects
- NSF funding model:
  - Supports projects upto 5 years. Software lifetime 20+ years
  - Expects community to support the software after NSF funding is over.
  - Software collaborations span countries, funding doesn't
- Transition to sustainability via practice (broadly speaking)
  - Incorporation into curriculum (and paid for by credit hours)
  - License fees or other revenues through commercialization
  - Open sourcing
  - Q: Is a technology push model viable?
  - Q: Does the community (which funded the software) retain use and an interest?

Challenges - Credit

- Metrics – How to measure software contributions, particularly in academic system?
  - Not just authors by order, but for all contributors
  - Need institutional buy-in, e.g., researcher metrics, P&T criteria
- Software Citations:
  - Dan Katz: *"I put some software on arXiv.org, and I got a URL. But this URL isn't quite the same as a paper's DOI. It is not indexed like a paper. Google Scholar, yes; Scopus & Web of Science, no. Is it curated and reviewed? Curated, yes. [Reviewed, no]"*
  - Pages not crawled by indexers do not appear in search results
  - Work with indexers: Products that are not indexed don't have their citations tracked => no credit
  - Need consistent metadata (see EAGER: https://github.com/mbjones/codemeta)
  - Need a curation and review process. Who will do it?