Building Computational Science through Sustainable Software : Experience and Perspective from the US National Science Foundation

Dr. Jennifer M. Schopf

United States National Science Foundation

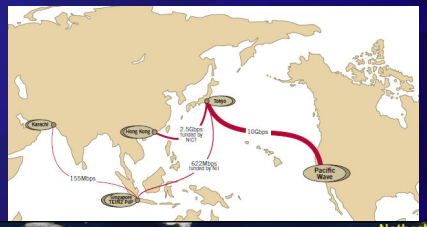
Office of CyberInfrastructure

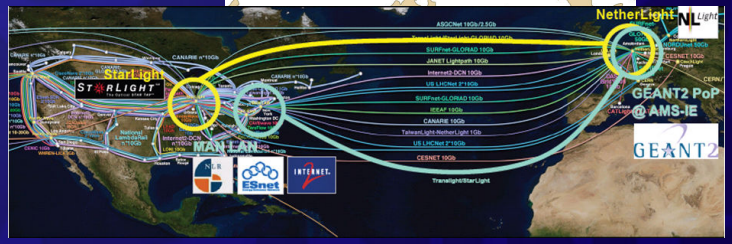
Sept 21, 2009

What Does Sustainability Mean?

- "Ability to maintain a certain process or state"
- In a biological context
  - Resources must be used at a rate at which they can be replenished
- Similar factors involved when looking at sustaining eScience infrastructure (cyberinfrastructure)

Sustainability in an Infrastructure Context: International Research Network Connections (IRNC)





- 2005-2010: $25M program
- 2010-2015: New $40M program
- All production awards must consider
  - Interoperability
  - Ongoing maintenance
  - Data sharing
  - Security
  - Sustainability beyond award
    - Significant leverage of other support

Sustainability in a Software Context

- Creating software that can be used in broad contexts
  - Reuse
- Funding models that encourage long-term support
  - Beyond normal funding agency grants

Note: I'm defining software VERY broadly – everything in your environment, middleware, tools, numerical libraries, application codes, etc.

The Problem: Software As Infrastructure

- Can we fund software sustainably the same way it does other infrastructure?
  - Same as telescopes, colliders, or shake tables
  - Line items in the directorate budgets
  - Constant or growing over time, reliably
  - Factor in "maintenance" and "replacement"
- Software lifecycle often longer than hardware
  - Hardware refresh ~3 years
  - Software can grow over decades

However, if software is viewed as infrastructure by funders then…

- Awardees must also treat it as such
  - Reliable, robust, reproducible, production-quality software
  - Reporting requirements (including uptime, usage statistics, and safety/security reporting)
  - Formal planning approach- including scheduling/estimation, requirements development, deployment plans, risk assessment, etc.
  - Teams with "professional engineering" backgrounds
- Change in culture for both development groups and funders

Note:

- This is not more money
  - More money isn't a solution here
- This is spending the money we have wiser

Outline

- We can have successes:
  - NMI program (MyProxy)
  - Lessons Learned
  - Software Development for CyberInfrastructure program
- NSF Encouraging Sustainable Software
  - Provenance
  - Education
  - Community Approach with Task Forces
  - Reuse

NSF Middleware Initiative (NMI)

- Established in 2001 to define, develop and support an integrated national middleware infrastructure
- "Making it possible to share scientific resources ranging from telescopes, supercomputing systems and linear accelerators to databases, directories and calendars."
- ~$12M 2001, amount varied yearly

Something That Worked: MyProxy (Welch, Basney; NCSA)

- Started at NCSA, 2000
  - Von Welch and Jim Basney
  - Provide an online credential repository for Grid portals and Globus Toolkit (GT)
  - Initial development from NLANR and NASA CORE, then NASA IPG provided first "sustaining" funding to support MyProxy for their use
- July 2002-June 2005 NSF Middleware Initiative (NMI) funding
  - NMI Grids Center (used NMI Build and Test)
  - Funded testing, hardening, documentation, packaging activities, bug fixes (and tracking)
  - Release process definition (and inclusion on GT)
  - Some development of additional features

MyProxy (cont.)

- Subsequent funding from
  - TeraGrid: Support its use in the project
  - NSF Dependable Grids ITR: MyProxy's failover functionality
  - NSF Strategic Technologies for CI (2009)
- Additional development and support from
  - European DataGrid, U. Virginia, LBNL, and others
  - Open source (and open contribution)

MyProxy Today

- Used by:
  - EGEE, EU DataGrid, Earth System Grid, FusionGrid, LHC Computing Grid, NASA Information Power Grid, NCSA, NEESgrid, NERSC, Open Science Grid, and TeraGrid
- MyProxy Usage:
  - TeraGrid: 21,744 requests from 775 users in July 2008
  - WLCG: 230,000+ requests/day

MyProxy: What Can We Learn?

- Satisfied a clear user need from the start
  - Expanded organically to satisfy users
- Built and maintained user confidence
  - Clear mechanism for users to communicate with the development team, and good documentation
- Maintained stability
  - Each release has always been backwards compatible

MyProxy and Open Source

- Coherent architecture, simple software design and open source
  - Basic prototype was stable and usable
  - Documentation strong from the beginning
  - Coordinated new features and contributions
  - External modifications and contributions are extremely cost effective

Open Source software is free…

Free as in speech…

free as in beer, or…

Free as in Puppy…

- Long term costs
- Needs love and attention
- May lose charm after growing up
- Occasional clean-ups required
- Many left abandoned by their owners
- May not be quite what you think

Additional MyProxy Lessons Learned

- End-user Involvement
  - Understanding user needs is VERY hard
  - Having a member of the user community work closely with the dev team is key
  - Most end users are not administrators
- Saying no (especially to features you *think* someone might like)
  - Over-selling, over-hyping software consistently backfires
  - "It's better to make half a product than a half-a$$ed product" – *Get Real*, 37signals

My Proxy Extras

- Today, MyProxy is distributed as part of the Globus Toolkit, the NMI GRIDS Center, Univa Globus Enterprise, and the Virtual Data Toolkit. MyProxy is used in many large grid projects, including the Computational Chemistry Grid, Earth System Grid, EGEE, FusionGrid, LHC Computing Grid, Open Science Grid, and TeraGrid.
- MyProxy recently underwent a security analysis by an independent third party: http://www.ncsa.uiuc.edu/News/09/0223MyProxypasses.html

Additional Lessons Learned

- Scope of the software plays a role
  - Do something, but not too big or too much
  - When it gets to be too complicated to be easily understood, well, no one understands it or uses it
- Smaller can be better
  - If you can only get funding for adding features, eventually you end up with something huge and unsupportable

A Harder Question: How to Choose What to Support

- "Everything should be made as simple as possible, but not simpler." –A. Einstein
- If we treat software as infrastructure, we have to pick *what* software to support
  - What is the REAL core of CI?
  - How do we have a coherent architecture?
- Will also need "exit strategy" as well
  - Eg. make it attractive for someone else (industry) to support

MyProxy Conclusion

- Use involvement is essential
- Clear, simple architecture and documentation
- Open source
  - And use external contributions
- Maintain stability
- Understand your core
  - In terms of software base, users, and deployments

Software Development for Cyberinfrastructure: SDCI (FY07)

- Develop, deploy, and sustain a set of reusable and expandable software that benefit a broad set of science and engineering applications
- HPC, Data, and Middleware target areas
- Required characteristics for proposals
  - Multiple application areas and expected usage
  - Awareness/distinction among alternatives
  - Project plan with proof-of-concept and metrics
  - Open source, use of NMI Build and Test (ETICS)
  - Demonstration in first 2 years

SDCI Outcomes

- 127 submitted proposals requesting > $145M
  - 26 awards, (5) HPC, (7) Data, (14) Middleware
  - Over $28M in total award funding
  - Co-funding support from BIO, ENG, MPS, and EPSCoR
- Some of the sw deployed and used in Europe
  - Inca (Smallen)
  - PerfSonar Framework (Swany)
  - SRB/IRODS (Moore)
  - OpenDAP/NetCDF (Gallagher)
  - BOINC (Anderson)
  - Condor (Livny)
  - Globus (Foster)
  - Pegasus (Deelman)
  - Kepler (Ludaescher)

Also Supported HPC and Data Tools

- IPM (Snavely)
- Cactus (LSU)
- Vulnerability assessment (Miller)
- Tau (Maloney)
- Parallel I/O (Choudary)
- Provenance collection (Plale)
- CalSWIM- data trust management (Lopes)
- Authorship attribution (Juola)
- Provenance Authentication (Jehani)
- Semantic Provenance (Fox)

Use of Software on OSG or TG

Software	Program	TeraGrid	Open Science Grid	EGEE
Condor	STCI	X	X	X
Globus	STCI	X	X	[parital]
GSI-SSH	SDCI/STCI	X	X	X
Inca	SDCI	X		[ngi's]
Metronome	SDCI		X	
MyProxy	SDCI/STCI	X	X	X
Pegasus	SDCI/STCI		X	X
SRB	SDCI	X		
NanoHUB	SDCI	X		
PerfSonar	SDCI			X
Vulnerability Assessment of Grid Software Infrastructure	SD CI			[parital]

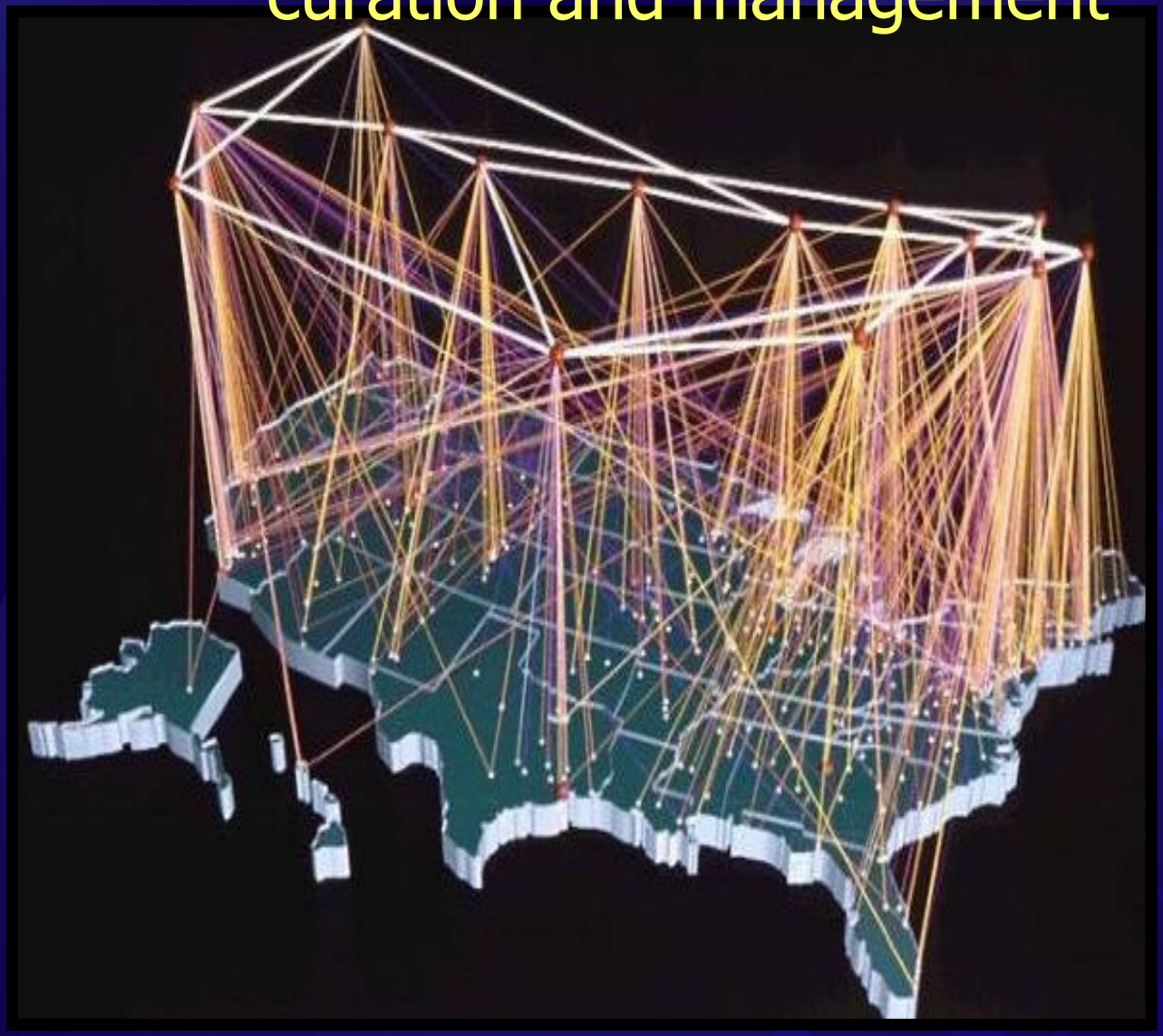
SDCI and STCI also supported data programs

- Data net

DataNet: Sustainable Digital Data Preservation and Access Network Partners Program

- Four primary goals:
  - Provide reliable digital preservation, access, integration, and analysis capabilities for science/engineering data over decades-long timeline
  - Achieve long-term preservation and access capability in an environment of rapid technology advances
  - Create systems and services that are economically and technologically sustainable
  - Empower science-driven information integration capability on the foundation of a reliable data preservation network
- Each project needed to develop a model for shared governance and the standards and protocols to enable interoperability

DataNet Vision: Partners Network of Networks to support all aspects of data curation and management



Data Net Projects

- Integration of library and archival sciences, cyberinfrastructure, computer and information sciences, and domain science expertise
- Work with multi-disciplinary science domains
- Engagement at the frontiers of computer and information science and cyberinfrastructure with research and development to drive the leading edge forward

2009 DataNet Awards

- DataNet Observation Network for Earth (PI: Michener)
  - Facilitates research on climate change and biodiversity, integrating earth observing networks
  - Emphasis on user community engagement, promote data deposition and re-use
  - Science question: What are the relationships among population density, atmospheric nitrogen, CO2, energy consumption and global temps?
- Data Conservancy (PI: Choudhury)
  - Integrates observational data to enable scientists to identify causal and critical relationships in physical, biological, ecological, and social systems
  - User centered design paradigm, ethnographic studies
  - Science question: How do land and energy use in mega-cities impact the carbon cycle and climate change?

Investment Levels

- Round 1: 2 awards of approx. $4M per year for 5 years
- Round 2: 3 awards of approx. $4M per year for 5 years
  - Preproposals evaluated, full proposals under discussion
- Co-funded by OCI ($80M) and CISE ($20M)
- Anticipate additional investment by other Directorates and Offices as the project yields demonstrable services

Investment Levels

- Round 1: 2 awards of approx. $4M per year for 5 years
- Round 2: 3 a… …r year for 5 years
  - Prepropos… discussion
- Co-funded by OCI ($…) … CISE ($20M)
- Anticipate additional investment by other Directorates and Offices as the project yields demonstrable services

Note: 5 year duration allows community planning

…but

- What else can we do to encourage and support sustainable software?

OUTLINE

- NSF Encouraging Sustainable Software
  - Provenance
  - Education
  - Community Approach (and Task Forces)
  - Reuse

Software Provenance

- Reproducible results are a requirement for basic science
  - In computational science, the science is in the code, data is in the code
- Software must be reliable and consistent
- Version tracking, metadata, environment tracking is critical

- Currently – a vast majority of computational science applications cannot be run by another researcher, and results cannot be reproduced

Teach Production Software Engineering

- One university's Software Engineering course
  - Systems analysis. Benefit/cost analysis.
  - Project scheduling, management, and control.
  - Requirements Specification document.
  - Development platforms. Prototyping.
  - Human factors. User interface design.
  - Detailed Design document. Configuration management. Program documentation.
  - Documentation. Installation. User training.
  - Software metrics. Cost estimation.
- Individual project developed during course of semester in addition

Software Engineering – for the applications people as well

- Address fundamental issues needed to work in a production environment
  - Working with a team – everything is more complex, from communication to version control
  - Working with end users – changing specifications, documentation, hand holding, negotiation
  - Operations – deployment, performance criteria, interoperabilty
  - Working to deadline – release requirements, tracking bugs, saying No!

Teaching Sustainable Software

- Care and feeding of production software
  - Understanding software life cycle
- Version control- software and practices
- Test (and build) frameworks
- Release management
  - Process for pushing a version out the door
- Documentation and communication
- Bug tracking
- Feature development with user interactions

Some Universities Do Teach These

- UCSD's SE course walks through Agile techniques
  - Work in a team with end users
  - Version control and release cycles
  - Weighing when to add features or harden

- But how many computational scientists would take this course?

Reuse

- In some ways, the best for of production quality code support
  - Mutual software adoption
  - Additional developers and users
- OCI funded "reuse" grants last year
  - Small amount of additional funds for existing awards to generalize, work with additional users

Reuse Example - HUBzero

- Started as NanoHUB
  - Support for nanotechnology software
  - Simulation framework for remote applications
  - Added content management system for educational content
  - Funded primarily out of the engineering directorate at NSF
- Request to generalize – OCI funded primarily
  - Created HUBzero framework
  - Generalized and simplified
  - Offered to other project groups

NanoHUB -> HUBzero

- Now in use by
  - MemsHUB- microelectromechanical systems
  - CCEHUB - cancer care engineering
  - ManufacturingHUB- advance manufacturing techniques
  - GlobalHUB - global engineering
  - PharmaHUB - pharmaceutical product development and manufacturing
  - ThermalHUB - heat transfer
  - IndianaCTSI – clinical, translational research (healthcare)
  - Center for Assistive Technology- asst. people with disabilities
- Offering HUB support in exchange for set yearly cost to other projects
- Formed the basis for bid for NEES Earthquake Engineering bid

Standards Adoption

- Fundamental to interoperability and reuse
  - Opens market
  - Allows competitive approaches
  - Allows easy replacement of components
- Another form of not re-inventing the wheel

CI Task Forces: Community Involvement in Implementing Vision

- Part of the Advisory Committee for CyberInfrastructure (ACCI)
- Community groups to help address holes in NSF CI vision document not yet addressed
  - And a new vision document is needed in a few years
- Get additional community input into OCI programs

CI Task Forces

Campus Bridging

Jennifer Schopf
jschopf@nsf.gov

Data (Viz)

Jon Stoffel
jstoffel@nsf.gov

Software

Manish Parashar
mparasha@nsf.gov

Rob Pennington
rpenning@nsf.gov

HPC (Clouds Grids)

Barry Schneider
bschneider@nsf.gov
Susan Winter
swinter@nsf.gov

Education Workforce

Rob Pennington
rpenning@nsf.gov

Grand Challenge VOs

Task Force General Strategies

- Timelines: 12-18 months
- Co-organized by NSF Program director and Advisory Committee for CI (ACCI) member
  - Membership from ACCI, community, other agencies (DOE, EU, etc.)
  - Involvement of NSF: OCI + other
- Workshop(s) and Recommendations
- We then go back and develop programs
- Areas: Software, Campus Bridging, Education/Workforce development, HPC, Data and Visualization, Grand Challenges and VOs

CI Task Forces and Sustainabilty

- All areas are dealing with sustainability to some extent
  - SW as infrastructure
  - Maintaining campus infrastructures and growing to national interactions
  - Data preservation (and policy)
  - Growing new computational scientists

Current List of TFs

- TF1 Software (M. Parashar)
  - Tools, compilers, appl frameworks, debuggers …
  - Software for comprehensive CI environments include networks, grids, clouds, datanet, etc
  - Community frameworks and toolkits for solving complex problems that may include all the above
  - Also: sustainability!
- TF2 Campus Bridging (J.Schopf)
  - What can we do to better integrate campus environments into regional/state/national CI
  - Networking, software stacks ,socio-political, etc.
  - Also: sustainabilty!

Current Task Forces

- TF3: Edu/WF development (R. Pennington)
  - Developing people who can do all this, from cs to sociology
  - K-20: Cyberlearning, teaching computational science and collaborative skills
  - REU and up: grad, postdoc, CAREER awards, computational science curriculum development
  - Also: sustainability!
- TF4 HPC/computing (R.Pennington)
  - More focused on the "hardware environment" roadmap, including petascale, exascale, grids, clouds
  - Also: sustainability!

Current Task Forces

- TF5 Data/Viz (J. Stoffel)
  - Going beyond DataNet, what do we need to do about the new data-driven science
  - Also: sustainability!
- TF6 VOs and Grand Challenges (B. Schneider)
  - Next generation grand challenge communities that may span disciplines, may use all the above to solve very complex problems.
  - And… (wait for it) sustainabilty!

Learning from Others: NASA Reuse WG

- Reuse Readiness Levels (RRLs) assess the maturity of sw products for potential reuse

1: No reusability; the software is not reusable

2: Initial reusability; software reuse is not practical

3: Basic reusability; might be reusable- skilled users, substantial risk

4: Reuse is possible; might be reused- most users, substantial risk

5: Reuse is practical; could be reused- most users, reasonable risk

6 Software is reusable; can be reused- most users, may be some risk

7 Software is highly reusable; can be reused- most users, minimum risk

8 Demonstrated reusability; has been reused by multiple users

9 Proven reusability; is being reused by many classes of users over a wide range of systems

- http://esdswg.gsfc.nasa.gov/WG/REUSE/index.html

Learning from Others: OMII: Open Middleware Infrastructure Institute

- Initial problem –provide hardening and sustainable funding for UK eScience program
- Grant awarded in 2 parts
  - Integration and packaging (similar to VDT)
  - Coherent hardening of software in community
    - Defined engineering process, open source license, code availability, etc.
- Pro: Engineering integration by project office
- Con: Effort required far exceeded resources; Early intervention needed for better results

Sustainable Software Policies?

- Many groups defining policies to preserve data artifacts
  - Data must be made publicly accessible
  - Data has to be stored in a national archive, etc.
- What about preservation of software?
  - Publicly accessible (more than a nod to open source)
  - Testing results available
  - Required demonstrations
  - Required end-user vouching?

Can we…

- Define metrics of use
- Define metrics of production software not research software
- Capture requirements on the software process (not requirements on functionality itself)
- Make use of professional developers more common place and accepted?

How do we deal with the academic versus production software conflict? How do we reward sustainable software?

Wrapping Up: What can be done within OCI and across NSF

- Sustainable approaches to software require a culture change
  - Longer term, predictable, and adequate funding streams
  - Production quality software, emphasizing solutions to real problems, reuse
- NSF policies and programs are attempting to encourage these changes
- Sustainability is one of the cross-program focuses of OCI

Some Questions We Can Try to Answer

- How can funding agencies encourage PIs to produce sustainable, reusable software?
- How can educators assist in the production of sustainable, reusable software?
- What are approaches that have worked well (or pitfalls to avoid) from our own track records?
- How can we encourage software written for one discipline to be able to be reused in another?
- What's the right ratio of software to hardware funding in a large-scale CI project?
- How should NSF decide what software to support?
- How do we build in rewards for producing sustainable software?
- How do we provide metrics for sustainability?

More Information

- More Information
  - Jennifer Schopf
    - jschopf@nsf.gov or jms@nsf.gov
  - José Muñoz
    - jmunoz@nsf.gov

- Thanks to:
  - Neil Chue Hong, Ian Foster, Peter Fox, Shantenu Jha, Miron Livny, Steven Newhouse, Ed Seidel (and the rest of OCI), Craig Stewart, Kevin Thompson, Alisdair Tullo, Von Welch, and many others