Sustaining Software Capabilities for the Long Term

OCI SDCI/STCI Workshop
January 29, 2010

Scott McCaulay – Indiana University
Director Research Technology Applications – UITS
Associate Director Digital Science Center - PTI
Cyberinfrastructure Software Sustainability and Reusability Workshop

• NSF-funded workshop March 27-28 2009
• 65 expert participants, diverse & international
• “...how is software that will be important to the US research and engineering communities identified, maintained, and supported over years to decades?”
• 12 findings and 14 recommendations
• Spirited discussion continues over the summary report, draft should be available
• Some overlap with our present conversation
  – Considerable focus on enterprise software
  – Science codes NOT considered CI software
• Don’t expect too many bold or controversial recommendations by consensus from a group this size
How much Track 1/Track 2 money should be spent on software and why?

• No “one size fits all” ratio
  – Experimental systems will necessarily have higher software costs than production systems

• What constitutes spending on software?
  – Porting? Optimization? Exclusively development?

• For Track 2A and 2B systems, <50% of the award went toward hardware acquisition (from press releases)

• Spending picture is further clouded by institutional match, which may support software

• Finding #5 - CI software generally has a longer lifespan than CI hardware
Evaluation criteria for software development and maintenance?

• Recommendations from CI Workshop include:
  – Formal documentation of user requirements
  – Appropriate software engineering practices
    • 10% of SE to meet 90% of needs
  – Licensed as open source
  – Usage metrics
  – Provenance and version tracking to ensure scientific reproducibility
  – Encourage platform portability if possible
• Recommendations generally echo language already in the SDCI solicitation, no radical new ideas
• Report encourages practices, not specific methods
“close collaboration with a scientific user community" or “broadly usable”

• CI Workshop emphasizes broad applicability
  – Sakai, Kuali
  – HUBzero Consortium

• Familiar and successfully sustained domain-specific codes were classified as “applications” rather than “infrastructure” and not considered by this workshop
Is there a system that would allow awardees to support utilized software?

- This already happens informally as PI’s include developers of critical software as partners or sub-awards.
- CI Workshop report notes the “grant mosaic” model of interlocking, intermingled grants as a successful model of software sustainability.
Standardization of software interfaces to enable interoperability?

• Probably a level of detail beyond the scope of the workshop, the report supports the concept without specific recommendations.
  
  – ...discussions on software interoperability and dissemination should be aggressively encouraged.
  
  – Sustainable software should use interfaces that do not depend upon extensive knowledge of their operating environment.

  – Software should be developed with clearly documented Application Program Interfaces (APIs).
Other Observations and Questions

• The workshop found a paucity of research on software sustainability. How can we know what factors really contribute to long term viability?

• Casual observation, anecdotal evidence show:
  – Extensively planned and well funded software failures
  – Naïve and spontaneous software successes
  – Successfully managed software projects are often flexible, dynamic, opportunistic and adaptive, not rigidly scheduled and executed

• Software development – art or craft?
  – Programmers are definitely not interchangeable parts
  – Are project management techniques designed for construction projects really useful or relevant for software development?
  – Given the high failure rate and erratic return on investment, the film industry may be a better model for software development 😊