Center For National Software Studies

Report on the Software R&D Workshop

...material for final report...

Bill Scherlis
Larry Druffel
Tony Jordano

29 Nov 04

Contents

- Background
  - Workshop focus
  - Why Software R&D
- Findings
- Hard questions
- Recommendations
  - Technical directions

Workshop focus

Center For National Software Studies

- Identify elements of a national R&D agenda for software.
  - Is there a significant strategic software challenge?
  - Are there ideas worthy of strategic investment?
  - Can we account for returns on the investment?
- Develop rudiments of a vision and strategy.
  - Will industry solve this problem for us?
  - Do we need a national software S&T strategy?

Findings, part 1

1. Software R&D has strategic significance
   - [See details on next slide.]

2. Industry R&D has significant gaps
   - The vast portion of industry R&D is focus on development activity in support of specific products and services
   - Generally speaking, industry lacks the incentive to create:
     - Nonappropriable foundational science
     - Pre-normative development of commonalities
     - Leapfrog revolutionary scientific improvement

3. IT is erroneously perceived to be at a plateau
   - Enterprise IT at a plateau (Harvard Business review)
   - Stake in the status quo: Reduced tolerance for disruptive change?
   - Poor quality products still accepted

Findings, part 2

4. In the long run, software engineering capability predicts system security
    - CERT: More than 90% of reported security events exploit software engineering flaws
    - What is pervasive is becoming critical
        - Beware of distinctions between critical and non-critical systems
        - We use the same engineering tools and practices
        - Availability is a function of component MTBFs plus ability to maintain security

5. Security and dependability appear to be high priority, but are not receiving investment
    - Much talk, but little action (cf. Boehlert hearing)
    - ROI case is still missing

6. Government appears to have withdrawn from leadership
    - Leadership is needed in addressing software engineering challenges (see, for example, funding levels in NITRD 2004 report)
    - Misinterpretation of statements such as, “It’s a management problem, not a technical problem”

Why Software R&D?

- Software is a strategic building material
  - National systems, business infrastructure, individual systems.
  - For U.S. national and economic security
    - Pervasive is becoming critical

- Software capability is a most significant differentiator
  - National economy and National security
  - Enable the next generation of IT-based innovation
  - Increasing challenge of the international competitive environment.
    - Loss of IT innovation leadership will create huge economic and security risks – Not like consumer electronics

- Significant advances in capability and quality are needed
  - Challenges in quality, dependability, security

- There are important recent advances
  - The advances are in multiple critical technical areas, and this work needs to be accelerated
    - Quality evaluation and assurance
    - Dependability and security
    - Frameworks and architecture-level assurance
    - Autonomous embedded systems
    - Mobile code and trusted remote execution
    - Advanced development practices and tools
  - There is broad new development of capable infrastructure
    - Software engineering tools and languages
    - Team-support capabilities
    - Frameworks, libraries, application creation

Recommendations

1. Create 2/5/10 year roadmaps for software R&D
   - Identify concrete software engineering Grand Challenges and Grand Strategies
     - And strategy to pursue them [See “R&D strategy challenges” slide, next.]
   - Joint government/industry/academic effort
     - Industry research leaders as active participants
     - An active federal research program helps industry innovate
   - Community assists in defining the “government business case” to engage
     - Research agenda must address the development of new measures – “good measures for what we care about”
   - Identify
     - Point of leadership
     - “Pulling apps” – e.g., national security, healthcare
     - Critical commonalities to stimulate – anticipate the pre-normative

2. Government should lead a strategic software R&D initiative
   - Focus on strategically significant technical areas
     - [See “technical directions” slide as a starting point for a planning process.]
   - Create assets: build a new community and shared culture
     - Without sponsorship, economics have forced academia and labs to disengage
     - Maintain strategic focus and direction
   - Attend explicitly to technology maturation
     - Mechanisms: Build and sustain testbeds, skunkworks, virtual laboratories, etc.
     - Bridging the gap from 30KLOC to 30MLOC
     - “Reality transition” – identify the driving problems

SWE R&D strategy challenges

- What are the elements of a business case for improved software capability?
  - Mission organizations: How do we place value on software capability?
  - National interest: How important is it to retain US innovation leadership for software R&D?
- How can industry, academia, and government work more effectively together?
  - What is the software industry?
  - How can public-private partnerships be crafted?
  - Is public-sector software research adequately funded?
- Is there a quality-related tip in the offing? (50yr point)
  - How can the pace of innovation be accelerated?
  - Many in the IT community believe this is likely, and will fundamentally shift market emphasis towards issues of quality, dependability, and security.

Technical directions

1. **Software structure, composition, and integration**
   - Beyond the limits of abstract data types and object orientation
     - New abstractions for abstraction
   - Software decay and continual refactoring
   - API interface design and its drivers
     - Compositionality
     - Frameworks, architecture, and patterns
   - Heterogeneous systems and composition
   - Domain specificity

2. **Models of design intent**
   - Elevating the level of abstraction
     - Better up-front assurances
     - Analysis-friendly design representation
   - "Attribute cross-cut" (aspect) analysis

3. **Direct evaluation / measures of software artifacts**
   - Models for non-functional attributes
   - Analysis at scale
   - Direct metrics for critical attributes of models, product, process
   - Substance, process utility, practicability

4. **Flexible process and team support**
   - Enabling process through tools and technology
     - Collaboration and multi-site development
     - Software design corpus: linking, rigor, assurance
   - Enabling iteration
     - Flexible process and aggressive measurement

5. **Software systems architecture**
   - Security, robustness, reliability as architectural attributes
   - Architecture for autonomy

6. **Software in systems engineering**
   - Software approaches to systems attributes: security, robustness, reliability
   - New architectures for distributed and embedded computing
     - Next-generation embedded operating systems
   - Concrete approaches to software for systems engineering