Some Critical Success Factors for Industrial/Academic Collaboration in Empirical Software Engineering

Barry Boehm, USC
(in collaboration with Vic Basili)
EASE Project Workshop
November 7, 2003
Outline

• Relevant (U.S.) CeBASE experiences
  – NASA High Dependability Computing Program
  – Army Future Combat Systems
• Some critical success factors
  – Incremental results
  – Sustained upper management commitment
  – CRACK participants
  – No missing links in adoption chain
  – Fully collaborative activities
  – Careful definition of data, metadata
  – Careful handling of intellectual property
• Conclusions
Relevant CeBASE Experiences
-U.S. context; may be different in Japan

- Government-sponsored collaborations
  - NASA High Dependability Computing Program
  - Army Future Combat Systems
  - NASA Software Engineering Lab
  - FAA Air Traffic Control Systems

- Direct industry collaborations
  - USC, UMD, FC-MD affiliate programs
Project Goal: Increase the NASA’s ability to engineer highly dependable software systems via the development of new techniques, processes, and technologies.

Research Goal: Develop high dependability technologies and assess their effectiveness under varying conditions and transfer them into practice at NASA


UMD/USC Level of Effort: $5 million over 5 years

Activities:

- Empirical investigation of NASA and NASA-contractor dependability problems
- Development of new technologies and engineering principles to address general forms of the problems
- Evaluation and iterative improvement of our results using realistic testbeds
- Model-based technology transfer which will provide the technology users with results of the effectiveness of the technology under varying conditions
HDCP Testbed Objectives

• Buy down risks of using new HDCP technologies
  – Pre-qualify new technologies in mission context
• Enable cost-effective HDCP technology integration
  – Dependability objectives vary by mission
  – Testbeds provide mission-relevant cost-effectiveness data
• Accelerate pace of HDCP technology maturity, relevance
  – Via early and accurate feedback
• Accelerate pace of technology transition
  – Usually around 18 years for software engineering technology
Accelerating Technology Maturity via Hierarchical Testbeds

- **Level 1**: Researcher-specific testbeds
  - Scenarios oriented around researcher’s technology
- **Level 2**: Common, distributable, mission-representative testbeds
  - Integrating Level 1 testbeds into common framework
  - Full complement of supporting capabilities
- **Level 3**: On-site, off-line mission testbeds
  - Test technology on actual NASA computers and software
  - Ability to use Level 2 supporting capabilities
- **Level 4**: On-site, live mission platforms and software
  - Carefully prepared; real proof of the pudding
SCRover Response to HDCP Testbed Criteria - I

• Representative of NASA, NASA-related missions
  – First external application of JPL MDS technology
  – Campus public safety robot
    • Using state-based autonomous control
    – Extensive review, support by JPL MDS personnel
• Full complement of supporting capabilities (current state)
  – Specs and code (UML, C++ baseline, xADL extension)
  – Mission scenario generations (MDS GEL-based)
  – Instrumentation (xADL/Mae assertion checks)
  – Tracers (seeded defects based on SCRover development)
  – Data analysis tools (xADL/Mae)
  – Experimental guidelines (FC-MD guidelines)
Defect Seeding

• Suppose HDCP technology finds 3 defects
  – Is this 100% of 3 defects, or 3% of 100 defects?
• Defect seeding
  – Seed testbed software with 10 defects
  – Suppose HDCP technology finds 6 of 10 seeded defects (60%)
  – Can estimate that it found 3 of 5 unseeded defects (60%)
• Assumptions
  – Seeded defects representative of existing defects
    • SCRover: obtained from project inspections, testing
    • Can also use representative NASA defect distributions
  – Test profile representative of operational profile
    • SCRover: use representative NASA mission scenarios
Example Intervention: xADL/Mae

• Refined SCRover UML specs into xADL*
  – Analyzed consistency, behavior with Mae tools
  – Instrumented code with xADL assertions

• SCRover testbed a good match for ADL interventions
  – Straightforward UML-xADL elaboration
  – Basic testbed infrastructure in place; usable for run-time assertion checking
  – Modest level of effort: 160 person-hours over 2 months

• xADL/Mae able to find 15 of 38 known defects, 6 unknown defects
  – Defect seeding analysis, defect distributions help determine what HDC techniques to apply next

• Successful comparative test of CMU Acme ADL
  – xADL and Acme found complementary defects
  – Led to NASA/USC/CMU effort to integrate, apply ADLs

* xADL: XML-based Architecture Description Language
Mae Defect Detection Yield by Type

- Interface
- Class/Obj
- Logic/Alg
- Ambiguity
- DataValues
- Other
- Inconsistency

Legend:
- #defects
- #Represented in Mae
- #Mae Detected

11/7/03
Applied Research: Army Future Combat Systems (FCS)

- Complex system of systems (CSOS): $4 billion for Increment 1
- CeBASE funded by FCS and OSD Software Intensive Systems
  - Third year: $1.2 million per year
- Members of SW Steering Committee and Program Office software support team
- Ensure software issues are addressed throughout the program
- Provide proactive expert consultation to the Program Office and integration contractor (Boeing)
- Collaborate with Boeing to apply risk-driven spiral model to software and system acquisition
- Capture and analyze empirical experience data to support downstream program decisions and future CSOS acquisitions
Future Combat Systems Risk Example: Limited speed of CSOS Software Development

– Many CSOS scenarios require close coupling of complex software across several systems and subsystems
– Well-calibrated software estimation models agree that there are limits to development speed in such situations
– Estimated development schedule in months for closely coupled SW with size measured in equivalent KSLOC (thousands of source lines of code):

\[ \text{Months} \approx 5 \times \sqrt[3]{\text{KSLOC}} \]

<table>
<thead>
<tr>
<th>KSLOC</th>
<th>300</th>
<th>1000</th>
<th>3000</th>
<th>10,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Months</td>
<td>33</td>
<td>50</td>
<td>72</td>
<td>108</td>
</tr>
</tbody>
</table>

Strategy to meet end-of-decade target (over 10,000 KSLOC):
– Use SAIV process. Architect for parallel incremental development, rapid integration of smaller supplier components
How Much Architecting Is Enough?
-A COCOMO II Analysis

Sweet Spot Drivers:
Rapid Change: leftward
High Assurance: rightward
Future Combat Systems Risk Example:
COTS Upgrade Synchronization and Obsolescence

Risk: Many subcontractors means a proliferation of evolving COTS interfaces

Risk: Aggressively-bid subcontracts can lead to delivery of obsolete COTS
- New COTS released every 8-9 months (GSAW)
- COTS unsupported after 3 releases (GSAW)
  An actual delivery: 120 COTS; 46% unsupported
Strategy: Contract provisions ensuring delivery of refreshed COTS products.
CeBASE CSOS Experience Base: Risks, Issues, Lessons Learned

• Building lessons learned experience base to
  • learn from early phases of FCS
  • improve later phases of FCS
  • provide an experience base for other DoD projects

• Example Experience Bases
  • An independent report of the top ten software risks as identified by the Software Team
  • A web-accessible software issue tracking system that captures select program issues brought to the attention of the software steering committee
  • A web-accessible lessons learned experience base that analyzes and synthesizes the software problem areas and tracks their evolution and resolution over time
Outline

• Relevant (U.S.) CeBASE experiences
  – NASA High Dependability Computing Program
  – Army Future Combat Systems

⇒ • Some critical success factors
  – Incremental results
  – Sustained upper management commitment
  – CRACK participants
  – No missing links in adoption chain
  – Fully collaborative activities
  – Careful definition of data, metadata
  – Careful handling of intellectual property

• Conclusions
Upper Management Commitment

• Personal and organizational commitment

• Stable sources of funding, key personnel, data

• Participation in reviews

• Responsiveness to problem situations
CRACK Participants

- **Collaborative**
  - Otherwise no teamwork
- **Representative**
  - Otherwise poorly-matched projects
- **Authorized**
  - Otherwise authorization delays or misleading "commitments"
- **Committed**
  - Otherwise missing participation, contributions
- **Knowledgeable**
  - Otherwise delays, unacceptable products

To get value from the collaboration, don’t send the people you won’t miss. Do send your crack (expert) people.
No Missing Links in Adoption Chain

- Avoid communication gaps
  - About technology, user domain knowledge
- Ensure rapid adaptation to change, problems
Fully Collaborative Activities

• Some co-location; some electronic collaboration

• Coverage of all adoption-chain links

• Co-evaluation of processes, tools, methods, metrics
  – Common core with special industry extensions

• Group prioritization activities
  – Stakeholder win-win negotiations
Careful Definition of Data, Metadata

• Common core with special industry extensions

• Management-relevant data
  – But not used in performance reviews

• Low data collection overhead
  – E.g, log file interpretation
Intellectual Property

- Data protection
- Data summarization
- Tool rights
- Non-disclosure agreements
  - Don’t overdo; don’t underdo
Conclusions

• Some definite successes and failures
• Critical success factors explain most differences
  – Incremental results
  – Sustained upper management commitment
  – CRACK participants
  – No missing links in adoption chain
  – Fully collaborative activities
  – Careful definition of data, metadata
  – Careful handling of intellectual property