Systems and Software Producibility
Collaboration and Experimentation
Environment (SPRUCE)

17 June 2010
Safe and Secure Systems & Software Symposium

Mr. Steven Drager
ACA CTC Lead
Information Directorate
Air Force Research Laboratory

DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.
PA Approval: #88ABW-2010-2781 dated: 25 May 2010
Outline

• Motivation and Vision
• SPRUCE Overview
• Sample SPRUCE Content
• What’s Next
• Concluding Remarks
The Problem

- DoD systems are inherently complex, increasingly software intensive
- Difficult to introduce advanced systems & software engineering technologies into DoD acquisition programs
  - Insufficient evidence to prove capabilities
  - Immature prototypes lack stability, features, support, tool chain integrations
  - Challenging to evaluate technology against “realistic” DoD problems
  - Must manage restrictions of classified, proprietary and ITAR information

- Increasing use of software: F-4 (8%), F-22 (85%), F-35 (est. >90%)
- Rapidly changing hardware speed and architectures
- Huge scale: $10^5$ requirements, $10^7$ lines of code
- Huge number of component interactions
- Increasing use of COTS and 3rd party software
- Long (20-40) year product lifecycles
- Stringent certification standards
Software-Intensive Systems Producibility Initiative

• Software is the prime enabler of complex weapons systems and command and control infrastructure

• Software is the least well understood and the most problematic element of large-scale systems
  – Little underlying science
  – Minimal engineering knowledge base

• Software and software project failures dominate causes of
  – System cost and schedule overruns
  – Failures of systems to satisfy their requirements
  – Increasing numbers of costly and dangerous system failures

• Research to improve our ability to develop complex software-intensive systems is token, disjoint, and narrowly focused
  – Industry has no incentive to solve common problems
  – Academia sees no consistent funding stream
• Systems & Software PRodUcibility Collaboration & Experimentation Environment (S²PRUCE² ⇒ SPRUCE)

• An open, collaborative research, and development environment to demonstrate, evaluate, and document the ability of novel tools, methods, techniques, and run-time technologies to yield affordable and more predictable production of software intensive systems
  – Brings together researchers, developers and domain experts from different communities to de-fragment the knowledge necessary to achieve SISPI research, development and technology transition

• Capabilities
  1. *Experimentation* on novel tools, techniques and methods
  2. Define *challenge problems* and *associated artifacts* to demonstrate the challenge problems
  3. Propose *candidate solutions* to challenge problems – i.e., technologies/tools/methods that address some or all parts of a challenge problem
  4. Define and conduct realistic *experiments* to assess candidate solution utility and document results and benefits to the DoD acquisition community
  5. *Collaborate* on all of the above
Root Cause

- **Ad hoc** collaboration within software engineering research community ⇒ “valley of disappointment” for DoD software research programs
  - Unable to find technologies that meet needs
  - Failure to adopt promising technologies
  - Software engineering problems encountered repeatedly across programs
  - Additional problem: “landing path” is not typically DoD programs
SPRUCE Vision

• Methods & technologies that create a common meeting house for program engineers & technology researchers to discover joint software producibility interests & form collaborations
  – With testbed resources & tools specialized to particular operational domains (e.g., avionics)
  – Driven by realistic artifacts
  – Bring researchers together with developers and development artifacts in an open collaborative research and development environment to ‘test drive’ (demonstrate, evaluate, and document) the ability of novel tools, methods, techniques, and run-time technologies to yield more predictable production of software intensive systems

• Benefit: challenge problem-driven, at-scale experiments can be defined & conducted to prove/disprove suitability of particular software producibility technologies
  – …which leads to a more systematic process for transitioning software producibility technologies
  – …enabling a more systematic process for technology transition
SPRUCE Usage Concepts

- SPRUCE portal designed with specific usage patterns, to make interaction natural for all the anticipated users of the portal.
- The idea was to use concepts that make it easy for users to create and navigate content, but that are not constraining.
- Communities of Interest can evolve naturally and organically from content, even after the content is in place.
- Key content is around Challenge Problems, Candidate Solutions and Experiments.
- Challenge Problems, Candidate Solutions and Experiments can reference each other, as shown.
SPRUCE Collaboration Environment

• Communities of interest: serve to organize SPRUCE content around broad but focused topic areas

• Challenge problems: sanitized versions of realistic problems that may occur on DoD acquisition programs
  – Past, current or future problems
  – Key: representative, sanitized artifacts

• Candidate solutions: describe proposed solutions to challenge problems

• Experiments and experiment instances: showcase challenge problems, evaluate solutions
What is SPRUCE (Physical)?

Customized Collaboration Portal + Repository of Realistic Problem Artifacts + Dedicated Experiment Resources
Experimentation Testbed

- Emulab infrastructure ([www.emulab.net](http://www.emulab.net))
  - Allocate testbed resources for research & experimentation
  - Configure allocated processors to run any supported operating system
  - Configure allocated network resources to different network topologies & behavior

- Current ISISlab infrastructure
  - 4 IBM BladeCenters & 7 Cisco switches
  - Per BladeCenter: 14 blades, 4 Gbps network I/O modules; 1 mgmt module
  - Per blade: two 2.8GHz Xeon CPUs, 1GB RAM, 4GB HD, 4Gbps network interfaces

- Multi-core equipment & software
  - HP Proliant DL140 G3 dual quad-core Intel Xeon E5320, 4GB RAM, 72GB HD
  - Dell PowerEdge dual quad-core Xeon, 32GB RAM, 2TB HD
  - Intel Performance Primitives 5.1
# SPRUCE Innovations

<table>
<thead>
<tr>
<th>Innovation</th>
<th>Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structured collaboration around specific challenges using experimentation</td>
<td>• Facilitates transition of promising technologies since stakeholders are already identified</td>
</tr>
<tr>
<td>and scientific exploration</td>
<td>• Addresses credibility gap by using demonstrations on program artifacts</td>
</tr>
<tr>
<td>Affinity matching of challenge problems, technologies and people</td>
<td>• Enables quick discovery and invitation of appropriate personnel</td>
</tr>
<tr>
<td></td>
<td>• Facilitates navigation and re-use of existing collaborations via affinity links</td>
</tr>
<tr>
<td>Seamless automation and linkage between SPRUCE collaboration portal and</td>
<td>• Enables parameterization, execution and automated results reporting, while using a sophisticated back-end experimentation infrastructure (Emulab)</td>
</tr>
<tr>
<td>experimentation testbed</td>
<td></td>
</tr>
</tbody>
</table>
Challenge Problems in SPRUCE

- A description of a software producibility deficiency that negatively impacts quality, capability, cost or schedule of software intensive DoD development programs.
  - e.g., assuring no race conditions in multi-threaded C++ or Java source code
- Organized into broad areas that contain one or more specific challenge problem instances
  - e.g., the problem described above could be a specific instance in the broad area of assuring non-functional run-time properties
- Types of challenge problems suitable for SPRUCE:
  - Type of R&D activity needed
    - Basic – focuses on fundamental scientific or technology gap
    - Applied – focuses on incremental capability gap or performance improvement beyond state of practice
  - Transition timeframe needed
    - Current generation programs – DoD acquisitions over next 1-5 years
    - Next generation programs – DoD acquisitions over next 5-10 years
Focus on Challenge Problem: Foundation for Success Criteria

• Challenge problems are the “killer content” that transforms SPRUCE from a web portal into a virtual community by:

1. Providing a conceptual foundation that attracts the primary groups – SPOs, contractors, researchers, & commercial vendors – in successful SISPI research collaborations

2. Providing real-world context, constraints & metrics to evaluate SISPI research progress

3. Providing well-defined transition & commercial tool opportunities for mature SISPI technology with proven results

Example: ATL QoS website
• Dedicated to the challenge of building distributed real-time embedded (DRE) systems from open QoS-enabled middleware, operating systems & networks
• Has attracted a participant community whose collaborations have driven middleware technology development
  – SPOs: DDG-1000, Aegis
  – Contractors: Boeing, Raytheon, Lockheed Martin, etc.
  – Universities: Vanderbilt, Wash U, CMU, Drexel, …
  – Commercial vendors: OIS, ZeroC, PrismTech, etc.

Example: Boeing Open Experimental Platform (OEP)
• DARPA PCES program
• Provided dozens of avionics system scenarios & software artifacts
• Drove evaluation of program composition technologies
• Ultimately established empirical evidence that enabled transition of ESCHER modeling tools onto FCS

Example: CMU’s Fluid Java code analysis tools
• Lockheed Martin’s Software Technology Initiative (STI) matched CMU’s mature Fluid Java code analysis technology to the U.S. Navy’s Aegis Open Architecture program & DDG-1000 programs
• Helped to establish business case for SureLogic startup & submission of congressional plus-up proposal between LM MS2-Moorestown, PMS 500, & SureLogic to apply SureLogic code analyzers to DDG-1000 R5 software QA
Collaborations in SPRUCE: Helping Problem Understanding

• Collaboration with problem contributors is critical
  – Challenge problem datasets are very large and challenges being faced are complex

• Examples from this case
  1. Are priorities present? Are they predefined and static? Dynamic?
  2. Can multiple tasks run at the same time?
  3. What type of scheduling is appropriate?
  4. How is task duration calculated?
  5. What are the scalability requirements?
Multi-Dimensional Resource Optimization

• Motivation: Avionics domain challenge

• Challenge: How to optimize allocation of processing tasks to processors, such that:
  — Processing tasks meet deadlines
  — Number of processors is minimized
  — Network bandwidth is minimized
  — Negative impact of internal bus I/O is minimized

• Artifacts: sanitized versions of avionics domain artifacts
  — Software workload
  — Network traffic
  — ~40 processors, ~100 software components and ~14,000 messaging interactions

• Problem and artifacts provided by Lockheed Martin Aeronautics
Sample Results from a Collaboration

Challenge problem comprised ~40 processors, ~100 components and ~14,000 messaging interactions

- 6 fewer processors needed than expected
- $13.9 \times 10^7$ Bytes = 24% network bandwidth reduction
- Solution found in ~9s on average
## SPRUCE Capabilities

<table>
<thead>
<tr>
<th>Capability</th>
<th>Purpose</th>
<th>Novelty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define Challenge Problems</td>
<td>Describe a software engineering deficiency that negatively impacts quality, capability, cost or schedule</td>
<td>“Killer” content that provides Foundation to attract participants Real-world context via artifacts Well-defined transition opportunities</td>
</tr>
<tr>
<td>Propose Candidate Solutions</td>
<td>Describe a potential solution to a challenge problem</td>
<td>Uses challenge problem artifacts as context for the solution</td>
</tr>
<tr>
<td>Define &amp; Conduct Experiments</td>
<td>Evaluate suitability/benefit of a particular candidate solution applied to a specific challenge problem</td>
<td>Contains experimental configuration and artifacts so that experiments can be reliably reproduced Includes results baseline, expected and actual results</td>
</tr>
<tr>
<td>Collaborate</td>
<td>Enable interaction among participants to refine content</td>
<td>Web2.0 technologies enable interactive refinement of problems, solutions and experiments</td>
</tr>
<tr>
<td>Match People, Problems &amp; Solutions</td>
<td>Get the right people with the right skills working together</td>
<td>Affinity metrics &amp; tools scan SPRUCE content and evaluate strength of relationships among people, problems &amp; solutions</td>
</tr>
</tbody>
</table>
Affinity Technology

• Purpose: provide automated mechanism to match people, problems and solutions based on SPRUCE content and people’s profiles

• Motivational scenario: challenge problem creation
  1. Create challenge problem
  2. Describe challenge problem with keywords – collection of word phrases
  3. Affinity technology will automatically suggest researchers with expertise to work on this problem

• Example affinities

<table>
<thead>
<tr>
<th>Affinity</th>
<th>Affinity is based on</th>
<th>Example Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Researcher to Researcher</td>
<td>Compatible interests, similarity of prior challenge problems worked on, similarity of tools used &amp; developed, ...</td>
<td>If you have worked with researcher A, you may find researchers B &amp; C’s work relevant</td>
</tr>
<tr>
<td>Researcher to Problem</td>
<td>How similar a problem is to the work created or used by the researcher</td>
<td>People that have worked on technology A are also likely to be conversant with Problems B &amp; C</td>
</tr>
<tr>
<td>Solution to Challenge Problem</td>
<td>How closely a solutions features match the components of a challenge problem</td>
<td>If you are interested in Problem A, perhaps solutions B &amp; C may be useful to you</td>
</tr>
</tbody>
</table>
• Approach: compare arbitrary keywords contained in SPRUCE content with profiles of researchers and research articles
  – CiteseerX scientific literature digital library database (hosted at PSU) provides content for researchers & research articles – contains ~1.5M publications
• Use probabilistic model to compute *affinity value*
  – Aggregate authors of publications into profiles for researchers
  – Create canonical set of researchers
    • For every text keyword, compute expected distance to all canonical researchers
      \[ E[d(k, \rho)] = \sum_{r \in S} P(k \in r) \cdot d(r, \rho) \]
    • Text keywords that are equally close to many canonical researchers are pruned as too ambiguous
    • Canonical researcher(s) that are closest to the given text keyword are assigned to describe that keyword
  – Citation graph distances between canonical researchers are used to assign distance between text keywords
Portal / Experimentation Testbed Integration

• Purpose: enable SPRUCE users to easily configure and execute/reproduce experiments
  – SPRUCE portal and experimentation testbed are two separate entities: different look-and-feel, functionality
  – SPRUCE experimentation testbed requires additional expertise that most SPRUCE portal users do not have

• Requirements
  1. Seamless integration between portal & testbed
  2. Secure access to testbed
  3. Ability to retrieve, display and select existing experiments from SPRUCE portal
  4. Ability to change parameters and launch experiments from SPRUCE portal
  5. Ability to automatically post experimental results to SPRUCE portal after experiment
  6. Consistent look & feel across portal, testbed
Portal / Experimentation Testbed Integration

• Introduce integration server to process requests and manipulate responses between SPRUCE portal server and testbed server

  ![Diagram showing integration between server and portal](image)

• Use SPRUCEbot to manage the data sharing between the SPRUCE portal and testbed
  
  — SPRUCEbot “owns” a SPRUCE testbed “project” under which SPRUCE experiments are created, displayed, launched, etc.
  
  — SPRUCEbot also responsible for testbed authentication and posting of experimental results to SPRUCE portal

• Use CSS from SPRUCE portal pages to maintain consistent look & feel
SPRUCE Program At A Glance

36-month program: April 2008 – March 2011

**Build**
- Standup SPRUCE; validate, refine & realize CONOPS
- Standup testbed infrastructure

**Populate**
- Populate SPRUCE with challenge problems, solutions, experiments
- Build initial SPRUCE user population

**Educate**
- Broaden SPRUCE awareness among participant communities
- User conferences, marketing, …

**Solicit**
- Increase SPRUCE adoption
- Challenge problem events, affinity metrics, evangelism, …

**Moderate**
- Sustain value
- Collect/post “value” metrics, maintain content freshness, …
# Representative List of Challenge Problems

<table>
<thead>
<tr>
<th>Challenge Problem</th>
<th>Provider</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache False-Sharing in Multi-Core Architectures</td>
<td>SPRUCE Team</td>
</tr>
<tr>
<td>Assessing the Evolution of Software Features</td>
<td>SPRUCE Team</td>
</tr>
<tr>
<td>Finding the Functionally Representative Areas of a Software System</td>
<td>SPRUCE Team</td>
</tr>
<tr>
<td>Multi-dimensional Resource Optimization for Publisher Subscriber-based Avionics Systems</td>
<td>SPRUCE Team</td>
</tr>
<tr>
<td>Advanced Multi-dimensional Resource Optimization for Publisher-Subscriber-based Avionics Systems Including Power Consumption</td>
<td>SPRUCE Team</td>
</tr>
<tr>
<td>Distributed Integration Strategies for Functional Components on Multi-Core Architectures</td>
<td>Univ. Arizona</td>
</tr>
<tr>
<td>SOA-MANET Distributed Sensors</td>
<td>Vanderbilt University</td>
</tr>
<tr>
<td>Technology Readiness Assessment Management Software</td>
<td>Edaptive Computing</td>
</tr>
<tr>
<td>Architecture Modeling</td>
<td>Raytheon</td>
</tr>
<tr>
<td>Software Quality</td>
<td>NIST</td>
</tr>
<tr>
<td>Correct-by-Construction Software Generation</td>
<td>Virginia Tech</td>
</tr>
</tbody>
</table>
SPRUCE Saplings

Federated SPRUCE Sites
- Allow multiple SPRUCE instantiations that give satellite sites control while allowing for smart sharing among the federation.

Cyber-Physical System Virtual Organization (CPS-VO)
- Actively support the formation of a multidisciplinary community and to facilitate broad-based collaboration on CPS.

Systems Engineering Research Center (SERC)
- Private collaboration and experimentation environment among the 20 universities

SPRUCE:
An open collaborative research & development environment to demonstrate, evaluate & document the ability of novel tools, methods, techniques & run-time technologies to yield affordable & more predictable production of software-intensive systems
- Customized collaboration portal
- Repository of realistic problem artifacts
- Dedicated experiment resources

Cyber Wargame Virtual Warfighter/Industry Collaboration-Environment

SPRUCE on the Front Line
Summary

• SPRUCE: an open, collaborative research, and development environment to demonstrate, evaluate, and document the ability of novel tools, methods, techniques, and run-time technologies to yield affordable and more predictable production of software intensive systems
  – Brings researchers together with developers and development artifacts to ‘test drive’ emerging technologies and techniques
  – Includes testbed resources and tools specialized to particular operational domains (e.g., avionics)
• Benefit: challenge problem-driven, at-scale experiments can be defined and conducted to evaluate particular software producibility technologies
  – Increasingly successful transition across the software producibility spectrum and across operational domains
  – Well-connected, self-sustaining community of participants
• For SPRUCE to be successful, WE NEED YOUR PARTICIPATION
Imagine a Future Where…

• As a program engineer toiling away, putting out day-to-day fires
  – You can easily discover & reach out to a broad community of people who can relate to your technical problems
  – You can easily learn about technologies that may help solve your problem
  – You can readily engage a community of experts to solve your problem
  – As icing on the cake, DoD research program managers are watching this activity, to solicit your input into their next project

• As a software researcher building exciting technologies and tools
  – You had a repository of real-world problems to work with
  – You can quickly & easily engage an active practitioner community
  – You had a means to effectively demonstrate how your technologies & tools can solve real-world problems

• As a research program manager/sponsor
  – You had access to a continuously evolving virtual community of people, problems & technologies from both constituents above
  – And that these constituents can contribute critical insights, data & artifacts to get your program justified, started & transitioned!
Revolutionizing the way that systems and software engineering technologies are identified, developed, and evaluated

www.sprucecommunity.org

Steven Drager
Air Force Research Laboratory/RITA
525 Brooks Rd.
Rome, NY 13441
TEL: +1.315.330.2735
Email: Steven.Drager@rl.af.mil

William McKeever
Air Force Research Laboratory/RITA
525 Brooks Rd.
Rome, NY 13441
TEL: +1.315.330.2897
Email: William.McKeever@rl.af.mil

Dr. Richard Buskens
Lockheed Martin
Advanced Technology Laboratories
3 Executive Campus, 6th Floor
Cherry Hill, NJ 08002
TEL: +1.856.792.9756
Email: rbuskens@atl.lmco.com