Reports from the Field: Evaluating the Readiness of Complex Software-Intensive Systems

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

John B. Goodenough
May 2, 2008

Software Engineering Institute | Carnegie Mellon

Summary

The SoS software assurance challenge

- Developing justified measures of confidence that (software-dependent) systems of systems will function as intended

This presentation

- Summarizes difficulties being faced in deciding whether to release complex systems for operational use
- Provides suggestive data on types of problems being experienced today in evaluating SoS readiness for use
- Describes relevant technical research problems and why they are important

In particular

- Results presented here are based on a series of interviews and visits with the stakeholder community

Overview

Background

The SoS Software Assurance Problem

Interview findings

Preliminary Conclusions

Background

SEI research project on software assurance

- Started in FY08

Research focus

- SoS assurance issues
- Measures of assurance

Overview

Background

The SoS Software Assurance Problem

Interview findings

Preliminary Conclusions

Software Assurance

The SAWG Definition of System Assurance

- Justified measures of confidence
- that a system functions as intended
- and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle

Software Assurance

- Software's contribution to system and SoS assurance
  - Software assurance in the context of a system's mission and use

Software Assurance

The SAWG Definition of System Assurance

- The justified measures of confidence
- that a system functions as intended
- and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle

Software Assurance

- Software's contribution to system and SoS assurance
  - Software assurance in the context of a system's mission and use
- Justified measures of confidence
  - Rational basis for deciding about SoS readiness for use
  - Evaluation in the context of how these measures are used to make decisions about fitness for use

Software Assurance

The SAWG Definition of System Assurance

- The justified measures of confidence
- that a system functions as intended
- and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle

Software Assurance

- Software's contribution to system and SoS assurance
  - Software assurance in the context of a system's mission and use
- Functions as intended
  - Involves user expectations, which change over time
  - Measures of confidence and levels of confidence change as usage changes

Software Assurance

The SAWG Definition of System Assurance

- The justified measures of confidence
- that a system functions as intended
- and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle

Software Assurance

- Software's contribution to system and SoS assurance
  - Software assurance in the context of a system's mission and use
- Environment of use
  - Actual environment of use (not just the expected environment of use)
  - Means evaluating robustness against unexpected use, threats, and changes in the environment

Software Assurance

The SAWG Definition of System Assurance

- The justified measures of confidence
- that a system functions as intended
- and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle

Our research assurance focus: mitigation of potential causes of critical failures in systems of systems, i.e., assuring SoS robustness

- Minimize impact of unusual (or unexpected) operational conditions
- Minimize impact of vulnerabilities that can be exploited by hostile entities
- Assurance that mission-critical functions are provided when required

SoS Software Assurance Research

Assuring the robustness of systems of systems despite

- Defects/vulnerabilities that are present
- Unanticipated usage or environmental conditions

Much current work focuses on defect prevention and defect removal

Our focus: assuring adequate SoS operation despite remaining defects

Research questions include defining:

- Architecture patterns to defend against failure modes arising from SoS interactions
- Methods of gaining confidence that SoSs are robust (testing is not sufficient)
- Appropriate measures of confidence
- Decision processes for releasing SoS requirements, designs, and implementations despite the existence of defects (both known and unknown)

Overview

Background

The SoS Software Assurance Problem

Interview findings

Preliminary Conclusions

Identifying Assurance Problems

One-hour Interviews

- T&E personnel
- SoS architects and designers

Approach

- Focus on large-scale or complex systems
- Explore successes/problems of today and concerns for future systems
- Follow-up with more detailed examination

Determine direction and transition barriers relevant for technical advances

Example Robustness Problems

1) An operationally deployed surveillance system
   - Brought up periodically and then shut down
   - When used continuously by another service, crashed after being up for an hour
2) Deployed program worked fine but
   - New users continually crashed the system
   - Had to use well-understood work-arounds
3) A system that stayed up long enough for demos, but not long enough for operational use
4) Changes in the field not consistent with new, sanctioned releases

Interview Findings (Highlights) (1/2)

Few critical deficiencies after release, but

- Need for work-arounds degrades released functionality
- Releases sometimes seriously delayed
- Problems growing as system interdependencies grow

Interoperability mismatches

- Some deviations from standards are critical; some are not. Which are which? How can they be detected earlier?

Release decision criteria

- Method for deciding when to release a system with known defects
  - Balancing impact of delay vs. actual operational impact of defects
  - Consideration of degraded modes of operation

Interview Findings (Highlights) (2/2)

Barriers to SoS assurance

- Deferred attention to mission-critical SoS failure modes
- Lack of basis for deciding what failure modes are best addressed at a given stage of the development process

Mission threads do not always address critical non-optimal mission conditions (e.g., impact of network failures, bandwidth limits, off-nominal loads)

- Non-functional behaviors are heavily dependent on the system and software engineering architecture, and these are not well-examined in the way mission threads are done today
- Functional requirements get the priority

SoS FMEA

Important SoS failure modes not always addressed in requirements

- Communication delays, resynchronization of networks
- Not always clear what failure response is appropriate, e.g., message ACK
  - Network load delays ACK message, causes resend, increasing network load
- Moving to a web-based architecture introduces new failure modes
  - In client/server architecture, migration away from servers can have severe impact on client performance (due to increased network delays)
- Tolerance for different operational doctrines, e.g., what data is loaded on a particular platform and who has access to it

New issues are discovered when system is released to field from OT

- Difference in usage environment

Standards and Interoperability

Representation of data values

- Float, double float, fixed point, number of bits, etc.
- Position values with respect to reference points
- Not adequately specified in interface descriptions

“Slight” deviations from standards

- Varied interpretations causing problems

Patch Management

How/Whether to distribute updates to field

- wireless, DVD, or what?
- Security patches vs. functionality patches?
  - Sometimes can't wait for next routine release
- Software updates/patches don't necessarily go through OT

Risk-based Release Decisions

Impact of deficiencies on warfighter

- Balance between impact of too many workarounds or unavailable capabilities vs. benefit of new capabilities that are working reasonably well

Impact of unspecified robustness requirements on cost and schedule

- When to address which failure modes
- Impact of deferring failure recovery requirements on ultimate system viability vs. impact of diverting effort to addressing failure modes that may not be that important in the end

Have SoS failure modes been addressed adequately?

- What evidence is needed?

Overview

Background

The SoS Software Assurance Problem

Interview findings

Preliminary Conclusions

Preliminary Conclusions (1/2)

Technical/Management Problems

- Failure to check for interoperation problems earlier than T&E
- Lack of consideration given to SoS failure modes and degraded modes of operation at the requirements level
  - Deciding on proper SoS response for various failure modes
  - Defining and considering off-nominal conditions and threads
- Release decisions for deployment based primarily on testing results
  - Sound decisions will need more information from development team

Current Approaches

- Testing robustness only if specified in requirements
- Moving toward earlier integration evaluation

Preliminary Conclusions (2/2)

Transition Barriers

- Spending effort to prevent problems that may never occur
- Incentives in favor of demonstrating capability under good conditions rather than acceptable performance under off-nominal conditions

Technical Gaps

- Methods for evaluating SoS robustness requirements, designs, and implementations to support release decisions
- Understanding SoS failure modes
- Evaluating stability of dynamically changing systems

Contact Information

Presenter: John Goodenough
Telephone: +1 412-268-6391
Email: jbg@sei.cmu.edu

U.S. mail:
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-2612
USA

World Wide Web:
www.sei.cmu.edu
www.sei.cmu.edu/contact.html

SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257

Acronyms

OMG	Open Management Group
SAWG	Systems Assurance Working Group
SoS	System of Systems
SEI	Software Engineering Institute
T&E	Test and Evaluation