Carnegie Mellon
Software Engineering Institute

Pittsburgh, PA 15213-3890

SIS Acquisition

Reconsidering the Role of Systems Engineering in DoD Software Problems

Grady Campbell (ghc@sei.cmu.edu)

Sponsored by the U.S. Department of Defense
© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Basic Questions

Are any software problems in DoD acquisitions traceable to systems engineering practices?

Would reformulating systems engineering and its relationship to software engineering reduce problems with software?

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Background/Motivation

The DoD focus on acquiring software-intensive systems

Lack of software guidance in systems engineering sources

Experiences in working with software organizations that information from systems engineering is often inadequate

Experiences with software product lines showing that most software product diversity traces to system-level information and tradeoffs

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

The Immediate Context

Systems engineering is the guiding technical approach for DoD acquisition programs

Systems engineering analyzes operational needs to define and create an enhanced operational capability

Systems engineering specifies required system behavior and decomposes the system into subsystems

Systems engineering is a primary conduit and filter for information to subsystem developers

Software engineering is seen as a specialized discipline focused on constructing software subsystems

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Significance

Any reasonably complex system has software as a critical element

Software is always part of a broader system

Systems and software decision making are interdependent and need shared visibility:

- Systems decisions constrain software alternatives
- Software decisions can significantly affect the emergent properties of a system

Systems engineering and software engineering need to overcome a conceptual incompatibility (physical versus information views of a system)

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Perceived problems for software

Lack of clear and complete information on needs

Failing to consider software properties and implications in system-level designs and trade studies

Software decisions ("after" systems engineering) that affect system properties

Systems engineering decisions that unnecessarily limit or complicate software alternatives

Poor fit of a top-down systems-software process to actual objectives (diverse or changing requirements)

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Relevance to DoD

DoD buys systems but software is both a critical enabler and a prominent source of risk (both product and process)

Systems engineering practices contribute to software risks if they:

- Prematurely over-constrain software engineering choices
- Inadequately communicate information, including unknowns and uncertainties, needed for effective software engineering
- Fail to adequately represent and analyze the implications of software design choices in system-level trade studies

Attempting to fix software engineering problems without rethinking the role of systems engineering may limit any potential for improvement

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

References

Systems Engineering Fundamentals, DoD Systems Management College, Jan. 2001.

Systems Engineering Capability Maturity Model® (1995) & *Capability Maturity Model Integration*® (2002), Software Engineering Institute.

R. Stevens, et. al., *Systems Engineering – coping with complexity*, 1998.

B. Thomé, *Systems Engineering - Principles and Practice of Computer-based Systems Engineering*, Wiley, 1993.

W. Wood, et.al., *DoD Architecture Framework and Software Architecture Workshop Report*, CMU/SEI-2003-TN-006, March 2003.

G. Campbell, *A Software Product Line Vision for Defense Acquisition*, CMU/SEI-2002-TN-002, June 2002.

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Questions for Further Investigation

- Are perceived problems of systems-software interdependencies and conflicts real and pervasive?
- Are there reasonable refinements or alternative reformulations of systems-software engineering that might reduce software problems?
- Could systemic reform of the systems-software engineering process and practices and applicable DoD policy produce improved results for acquisition?

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Some Goals for System-Software Interdependence

- System-software requirements and designs are iteratively refined
- Systems engineering applies software expertise to identify and evaluate alternative system decompositions
- Estimations of system attributes account for how software decisions can affect emergent properties
- Human interface requirements reflect current software capabilities
- Information on uncertain and changing needs is communicated to guide designing for change

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Prospective Reformulations of Systems-Software Engineering

(or transitional stages)

- Improve systems engineering performance by refining current methods & maturing their use in practice (e.g., CMMI®)
- Apply software practices in systems engineering to identify and analyze alternatives and system-level implications
- Closely integrate and iteratively apply systems and systems-level software engineering for analysis and design
- In systems engineering, build a system first as an abstract computation over an information model of the enterprise – substitute hardware for software as needed to optimize and deploy

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

Related, Advanced Topics

Process improvement

Hardware-software co-design

Product lines and mass customization

Modeling and emulation of hardware in software

© 2004 by Carnegie Mellon University

Carnegie Mellon
Software Engineering Institute

A Call for Discussion and Action

- Share perspectives and perceptions of software-intensive systems engineering
- Report on alternative system-software relationships in theory or practice
- Pursue answers to ‘questions for further investigation’
- Initiate efforts to collaboratively reformulate systems engineering for SIS or revise DoD policy accordingly

© 2004 by Carnegie Mellon University