Architecture Tradeoff Analysis: A Disciplined Approach to Balancing Quality Requirements

Dr. Azad M. Madni
Chief Executive Officer
Intelligent Systems Technology, Inc.

Working Group 4A
Architecture-Centric Evolution (ACE)
of Software-Intensive Systems

March 27, 2007

3250 Ocean Park Blvd., Suite 100, Santa Monica, CA 90405
310-581-5440 Fax: 310-581-5430 www.IntelSysTech.com

Outline

- Problem
- Architecture Tradeoff Analysis
- Common Misconceptions
- Findings
- ATA for SoS
- Promising Research Thrusts

Problem

- Software-intensive systems continue to grow in size, functionality and complexity
- Architecture increasingly being viewed as a tool for making decisions:
  - *operational community*: doctrine development and analysis
  - *acquisition community*: budgeting and planning
- How does one determine whether or not a proposed architecture satisfies competing "quality" requirements without implementing it first?

Need an architecture tradeoffs analysis methodology and toolset

Architecture Tradeoff Analysis (ATA)

- A disciplined approach for:
  - eliciting stakeholders' *quality requirements*
  - identifying and prioritizing *critical quality attributes* [Madni, et al., 1981, Madni, 1983]
  - uncovering and mitigating risks arising from design decisions that could lead to *quality problems*
- Strives to harmonize competing quality requirements that drive system architecture (e.g., performance vs. dependability)

> Can rarely simultaneously optimize for all quality requirements….hence the need for tradeoffs

ATA Characteristics

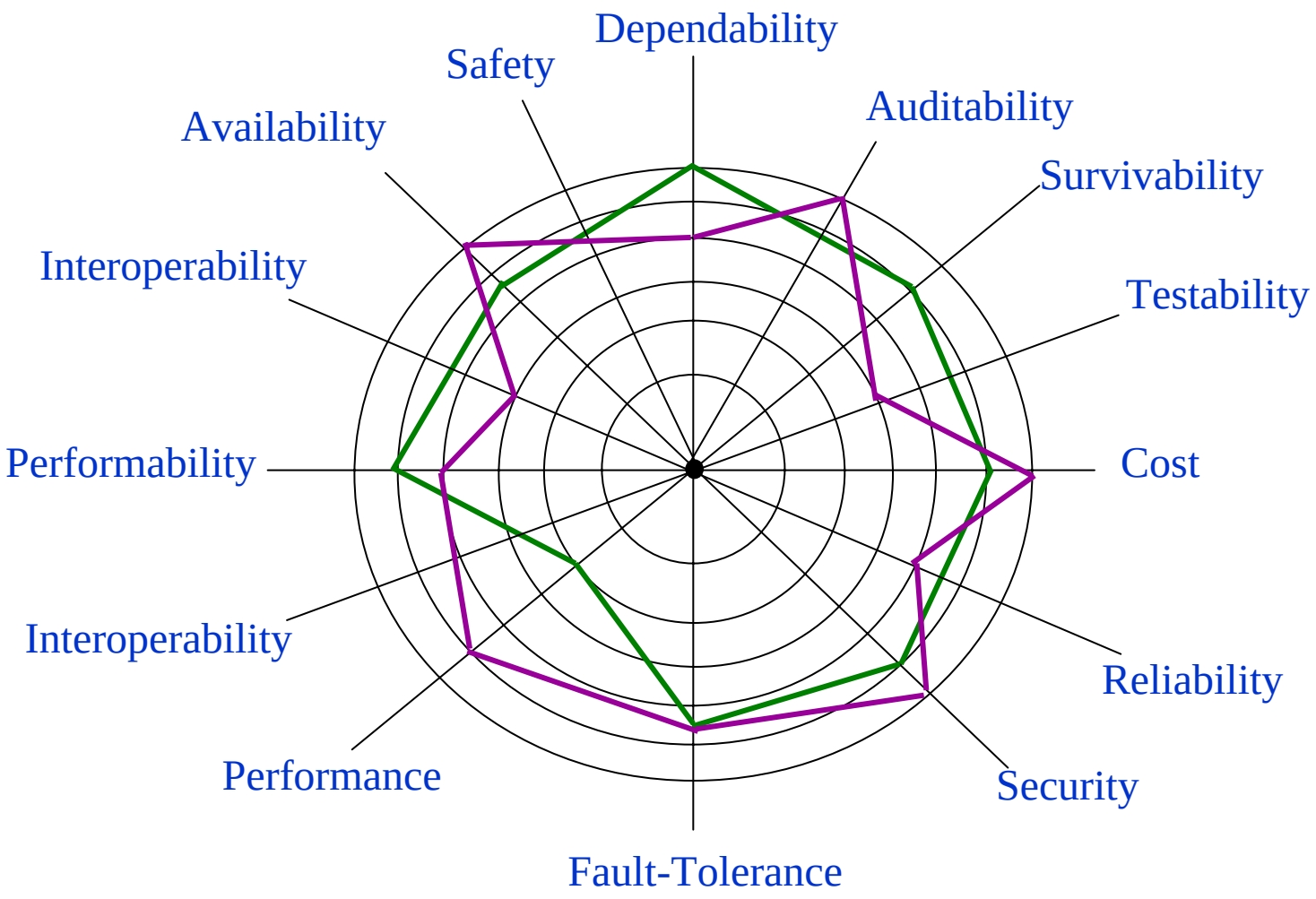
- Begins with ballpark estimates which are progressively refined
  - multi-resolution analysis including probing deeper using techniques such as benchmarking and prototyping
- Makes tradeoffs explicit and visible to architects [Madni, 1983]
  - minimizes the risk of not meeting quality requirements
  - quantifies impact of architecture decisions on quality requirements
- Architectural decisions affect interactions among quality attributes [Barbacci, et al., 1995]:
  - *sensitivity point:* applies to those decisions that enhance/degrade at least one quality attribute
  - *tradeoff point:* is a sensitivity point between two or more quality attributes that interact in opposing ways

ATA Benefits

- Facilitates identification, elicitation and definition of quality attributes and their requirements
- Improves communication among stakeholders
- Provides a basis for making sound architectural decisions
- Provides a basis for documenting architecture design decisions [Clements, et al., 2005]
- Facilitates identification of risks early in the system lifecycle

Comparing System Architectures Based on Quality Attributes

[Madni, 2007]



Common Misconceptions

(about architecture)

- Modularity Trap
- Architectural Complexity Reflects Problem Complexity
- Process Maturity Assures Quality
- Complexity Can Always Be Reduced
- Complexity Can Only Be Managed
- System Complexity Determines UI Complexity
- Decomposability is Synonymous with Modularity

Modularity Trap

- Modularity is viewed as a means to simplify design, manage complexity and improve maintainability
- Actually, modularity has its own problems
  - achieving “coherent connectivity”
  - creating appropriate definitions of mediating standards between modules
- Can have unintended consequences for maintainability and evolvability
  - unclear what level of modularity produces an effective tradeoff between “maintainability” and “coherent connectivity”
  - can potentially limit future innovation, resulting in incremental improvements within modules

Modularity is not enough!

Architectural Complexity Reflects Problem Complexity

- Actually, architectural complexity can be the result of human design decisions
  - unintended or accidental consequences of one's design ("design-induced")
- ATA allows architects to analyze/understand nature of architectural complexity
  - essential vs. accidental vs. optional complexity
  - man-machine interface complexity vs. implementation complexity
- ATA enables architects to document/analyze consequences of accidental or optional complexity

Avoid "featuritis"

Process Maturity Assures Quality

- Mature processes can control, predict schedule and cost but do not guarantee meeting other quality requirements
  - quality means different things to different people (e.g., CAIV, SAIV)
- Cannot collectively maximize all quality attributes

A mature process can only go so far!

Complexity Can Always Be Reduced

- Only design-induced architectural complexity can be reduced
- Systemic (structural) complexity, which is intrinsic to the problem domain, cannot be reduced

Avoid unwarranted optimism

Complexity Can Only Be Managed

- True for systemic complexity
- Not true for design-induced architectural complexity (e.g., overly complex or extraneous protocols) which can be reduced through, for example:
  - design/ protocol simplification
  - elimination of extraneous features

Avoid unwarranted pessimism

System Complexity Determines UI Complexity

- Much of system complexity can be concealed by the use of composable, context-sensitive displays, and automation
- The challenge is to make sure that decision-relevant information is not inadvertently left out

Avoid leaving out decision-relevant information during UI simplification, contextualization

Decomposability is Synonymous with Modularity

- Decomposability is a special form of modularity
  - a fully decomposable system is a modular system with no interactions among the modules
- A modular system (i.e., one with distinct sub-assemblies or components) often embed complex interactions within and/or among modules

Be mindful of interactions

Findings (stakeholders)

- Scenario-based design promotes understanding and consensus-building among stakeholders
- For stakeholder "buy-in," make evaluation goals clear
- Culture influences how scenarios are elicited from stakeholders and prioritized

Findings (architecture)

- One can learn as much from building models as analyzing them
- Architecture definition limits the levels of achievable quality
- Documenting architectures promotes understanding
- 25% increase in problem complexity can result in 100% increase in solution complexity [Glass, 2002]

Findings
(quality attributes)

- Quality attributes tend to interact and are, sometimes, in conflict
- Usability consists of “learnability” and efficiency
- Early design decisions are primary drivers of system quality
- Quality attributes often need to be traded off against each other to achieve desired system goal
  - e.g., replicating communication/computation to enhance dependability can have a deleterious impact on performance
  - e.g., co-locating critical processes to enhance performance might diminish dependability (e.g., single point failure)

Complexity in SoS

- Structural and/or emergent complexity
  - inevitable in large-scale and ultra-large-scale systems
  - result of substantial interaction among networked components
- Engineered complexity
  - an unintended consequence of human design decisions
  - stem from designed or accidental nonlinear and cyclic interactions between protocols and other architectural elements within a single network environment
  - can be controlled (e.g., design simpler protocols, resist "featuritis," simplify architecture)
  - Kolmogorov complexity useful to understand this type of complexity

ATA/SoS Research Needs

[Madni, 2007]

- How do we create a "glass-box" process for architectural tradeoff analyses
  - explicit identification of SoS quality tradeoffs
  - illuminate SoS architectural risks through identification of both quality attribute values and attribute trends
- How do we pinpoint and mitigate areas of potential risks
- How do we uncover where and which quality attributes (and capability) are affected by specific architectural design decisions for the different SoS configurations

ATA/SoS Research Needs (cont.)

[Madni, 2007]

- How do we define quality attributes for a SoS
  - in SoS, quality requirements pertain to the overall ensemble behavior; i.e., local interaction must collectively render the required SoS-level quality attributes (local actions resulting in desired global properties/behaviors)
- How do we treat legacy systems as functional invariants when performing tradeoffs
  - ability to distinguish between "as-is" and "to-be" during tradeoff analysis
- How do we accommodate the dynamic and fluid nature of SoS when performing tradeoffs
  - emphasize tradeoff analysis with respect to predictability, survivability, scalability and robustness in light of legacy systems/capabilities

"Everything should be made as simple as possible, but no simpler."

- Albert Einstein

Azad M. Madni
amadni@intelsystech.com

References

- Barbacci, M., Klein, M., Longstaff, T., and Weinstock, C. Quality Attributes, CMU/SEI-95-TR-021, December 1995
- Glass, R.L. Sorting Out Software Complexity, Communications of the ACM, Vol. 45, Issue 11, pp 19-21, November 2002
- Madni, A.M., Moini, A., and Madni, C. Towards an Architecture Tradeoff Analysis Framework for System-of-Systems, accepted for publication in the *Journal of Integrated Design and Process Science*, 2007
- Madni, A.M. Architecture Tradeoff Analysis: Towards a Disciplined Approach to Balancing Quality Requirements, USC-CSSE Executive Workshop, Annual Research Review Presentation, February 14, 2007.
- Madni, A.M. Integrated Modeling Approaches in Advanced Cockpit Automation, *Proceedings of the 1983 SAE Aerospace Congress & Exposition*, October 1983
- Madni, A.M., and Freedy, A. Decision Aids for Airborne Intercept Operations in Advanced Aircrafts, *Proceedings of the 1981 International Conference on Systems, Man, and Cybernetics*, pp. 224-234, 1981

Azad M. Madni, Ph.D.
Chairman and CEO, ISTI

- President, Society of Design and Process Science (SDPS)
- Editor-in-Chief, Journal of Integrated Design and Process Science
- Fellow of IEEE, INCOSE, SDPS
- Associate Fellow of AIAA
- Developer of the Year in 2000, 2004 at Software Industry Awards
- 2006 C.V. Ramamoorthy Distinguished Scholar Award from SDPS for seminal contributions to design and process science
- Selected by DARPA IPTO for *Sustained Excellence by a Performer* and *Significant Technical Achievement* Awards at DARPATech 2004
- SBA's 1999 National Tibbetts Award for California (innovation, entrepreneurship)
- Mass Mutual and Chamber of Commerce 2002 Blue Chip Enterprise Award
- Several awards and commendations from DARPA, OSD, and Navy for innovations in concurrent engineering, agile manufacturing and human-system integration
- Principal Investigator on R&D projects sponsored by: DARPA, HSARPA, OSD, MDA, AFRL, AFOSR, NSWC, ONR, NAVSEA, NAVAIR, NRL, CECOM, AMCOM, RDECOM, ARI, HEL, MARCOR, NIST, DoE, and NASA