



Breakout Session 8A

Architecture-Centric Evolution (ACE) of Software-Intensive Systems

Session Chairs

Sergio Alvarado, Ph.D.

Michael Hogan

Phillip Schmidt, Ph.D.

The Aerospace Corporation

John Georgas

UC Irvine Institute for Software Research

ACE Session Goals:

Software Architecture Recommendations

- Develop **recommendations** on appropriate **level of granularity** of software architecture needed to support evolution of software-intensive spacecraft ground systems
- Attempt to address multiple aspects of software system evolution
 - Changing requirements
 - Evaluation
 - Development
 - Implementation
 - Integration
 - Testing
 - Verification
 - Maintenance

ACE Session Methodology: Gain Consensus on Recommendations

- Aerospace Corporation developed **ACE session baseline**
 - **Discussion statements** on software architecture for system evolution
- Invited presentations to discuss, refine, and gain consensus on baseline
- Presentations organized in three discussion segments
 - **Architecture Requirements for Evolution and Evaluation**
 - **Architecture Methodology and Software System Lifecycle**
 - **Architecture Methodology and COTS-Based Systems**
- Discussion segments included three or four presentations and a question answering period
- ACE session concluded with a synthesis segment for formulation of session's findings

ACE Session Presentations: Architecture Requirements for Evolution and Evaluation

New Roles for Architecture

Maj. C. Beres

Air Force Space and Missile Systems Center

Architecture: Acquisition's Holistic Medicine

Capt. M. Raphael

Air Force Space and Missile Systems Center

Architecture-Centric Representation for Design Diversity and
Program Evolution

P. Schmidt, S. Alvarado, J. Rivera, and J. Milstein

The Aerospace Corporation

ACE Session Presentations: Architecture Methodology and Software System Lifecycle

Usability Constraints on Architecture Development and Use

J. Reeves

Northrop Grumman Mission Systems

A Focused Approach to Software Architectural Recovery

N. Medvidovic and V. Jakobac

USC Center for Software Engineering

Recommendations for Architecture-Centric Software

Supporting Self-Adaptive Behavior

J. Georgas

UC Irvine Institute for Software Research

ACE Session Presentations: Architecture Methodology and COTS-Based Systems

An Adaptable and Market-Driven COTS Command and Control Architecture

S. Norcross

L3 Communications Corporation

Deploying a Common Software Architecture for Real Time Launch, Test and Satellite Ground Systems

R. Andzik

Real Time Logic Inc

Balancing Generic Software Component Design with Tailored COTS Solutions

M. Gilmore

Raytheon (Aurora Campus)

Architecture Granularity – A COTS Vendor's Perspective

B. Grasso

Integral Systems Inc

ACE Session Baseline:

Overview of Discussion Statements Given to Presenters (1)

1. Architecture as Blueprint for Evolution*

Integral part of software system evolution rather than a documentation afterthought

2. Multi-View Architectures*

Software system evolution dependent on architecture models with concurrent multiple views: logical organization, dynamic behavior, software organization, process decomposition, and physical realization

3. Architecture as Decision-Making Tool*

System evolution decisions made by addressing front-end conceptual issues versus back-end implementation issues

4. Architecture Representation*

Electronically represented/shared using UML (Unified Modeling Language) or an ADL (Architectural Description Language)

5. Component-Based Architecture*

Organized in terms of software components with well-defined interfaces that encapsulate and provide access to component functionalities

*Focus of discussion

ACE Session Baseline:

Overview of Discussion Statements Given to Presenters (2)

6. **Architecture as Basis for Requirement Verification**
System requirements traceable to detailed component representation
7. **Architecture as Basis for System Testing**
Test plans and system failures traceable to detailed component representation
8. **Architecture as Basis for System Implementation***
Mapping detailed component representation into COTS products, developed code, or both
9. **Architecture as Tool for Managing COTS-Product Change**
Component-to-COTS mapping and access to COTS-products' internal structure provide basis for product upgrade or replacement
10. **COTS-Product Independent Architecture**
Independence from specific COTS products selected for implementation ensures generic framework for COTS-product integration

*Focus of discussion

ACE Session Findings: Architecture as Blueprint for Evolution

- **Baseline**
 - Integral part of system evolution
- **Consensus**
 - Software architecture is a core asset
 - » Needed to reduce risk, manage complexity, and inevitable change
- **Discord**
 - **Granularity level**, ownership, and responsibility for software architecture

ACE Session Findings: Multi-View Architectures

- **Baseline**
 - Software system evolution dependent on architecture models with concurrent multiple views
- **Consensus**
 - Usefulness of multiple architecture views
 - Availability of various methodologies for representing views
- **Discord**
 - Appropriateness of architecture representation methodologies
 - **Granularity level** of architecture views

ACE Session Findings: Architecture as Decision-Making Tool

- **Baseline**
 - System evolution decisions made by addressing front-end conceptual issues versus back-end implementation issues
- **Consensus**
 - Provides long-term means for stakeholders communication
- **Discord**
 - Integration of architecture as a decision-making tool into all software system evolution processes
 - » Period of usability of architecture as a decision-making tool
 - » **Granularity level** of architecture representation to enable independent assessment by each stakeholders

ACE Session Findings: Architecture Representation

- **Baseline**
 - Electronically represented/shared using UML (Unified Modeling Language) or an ADL (Architectural Description Language)
- **Consensus**
 - Need for tools for representing software system architectures
- **Discord**
 - How to determine adequacy of contractually-delivered architecture representations
 - » **Granularity level** of UML or ADL representations
 - » How to improve architecture representation

ACE Session Findings: Component-Based Architecture

- **Baseline**
 - Organized in terms of software components with well-defined interfaces that encapsulate and provide access to component functionalities
- **Consensus**
 - Need for developing and managing components
- **Discord**
 - **Granularity level** and availability of component representations
 - » Dependent on **proprietary concerns**
contractual issues (cost and schedule)
technical issues (component functionality and interfaces)
requirements (performance, reliability, and service quality)

ACE Session Findings: Architecture as Basis for System Implementation

- **Baseline**
 - Mapping component representation into COTS products, code, or both
- **Consensus**
 - Need for mapping component-based architecture to implementation
- **Discord**
 - **Granularity level** of the mapping
 - » Dependent on **proprietary concerns**
 - contractual issues** (cost and schedule)
 - technical issues** (component functionality and interfaces)

ACE Session Summary

- Forum for software-system experts, users, developers, and researchers to discuss views on software architecture granularity for system evolution
- Establish consensus on what issues need to be addressed
 - Architecture as a core asset
 - Usefulness of multiple architecture views
 - Architecture as a long-term means for stakeholders communication
 - Tools for representing software system architectures
 - Component development and management
 - Mapping component-based architecture to implementation
- Uncovered need to bridge stakeholder differences
 - Need for guidance on how to determine and agree on appropriate software architecture granularity level

Recommendation:
Make Architecture-Centric Evolution the Main Theme for GSAW2004