Overview of ACE2
Presentations

Thomas A. Alspaugh
Institute for Software Research
University of California, Irvine
31 March 2004
Lt Col Laura Pope
(Air Force Space and Missile Systems Center)

- Better architecture up front => better system
- System architecture, not software architecture
- Architecture should be the model for evaluation
  - Consider operations, maintenance
  - Address issues up front
  - Neither requirements nor code are right level
- Requirements are never adequate
- Addition: scenarios of use that express tests
Dr Joel Sercel
(MILSATCOM Joint Program Office)

- Understanding is important, not architecture
  - Good understanding precedes good architecture

- Architecture = set of constraints on designs

- Choose constraints that are effective
  - in achieving the qualities you need
  - example: invariants aid change management

- Necessary for managing change
Dr Linda Northrop
(SEI)

- SA is structure(s) comprised of
  - software elements
  - their external behaviors
  - the relationships among them

- Architecture is the center of many activities
- Scenarios are more expressive than attributes
- SEI has a number of SA techniques and methods
- All the ACE2 objectives are quality attributes
Dr Peter Hantos
(Aerospace)

- The system is what is important
  - Architecture is just a way to achieve system goals
- An architecture is a dynamic entity that evolves
- Architecture-centric development process covers long list of aspects
- Use cases bind all the core workflows together
- Don’t use MIL-STD-1521B
Overall
(1st session)

- What is architecture?
  - set of constraints
  - components, behaviors, relationships
  - ...
- System architecture or software architecture?
- What can architecture do for you? Everything?
- When / how are scenarios useful?
- Good architecture precedes good system
  - What is a good architecture?
  - What precedes a good architecture?
Capt Bryan Berg
(Air Force Space and Missile Systems Center)

- Architecture: a “string” to perform a contact
  - ~5 components, their functions, and their interconnections (in terms of SEI defn)
- COTS components + in-house “glueware”
  - “glueware” isolated COTS component changes
- Upgrades difficult (except one case)
  - No control over COTS component evolution
- Plan to use industry standards to ease upgrades
Peter Shames (JPL)

- UML-based reference architecture for space data systems
- Several views of system
  - each with its own kinds of components and connections
- Its use: describe (model) the system, then reason using the description
- Primarily addresses understandability
  - maintainability, extensibility, executability indirectly
Jim Boegman  
(Raytheon)

- Architecture is higher-level view than design
  - architecture above design above implementation
- Requirements at all these levels
- They find architecture (in this definition) is insufficient to assess maintainability, etc.
  - More detail is needed, such as a prototype
Dr Allen Nikora.
Myron Hecht, Douglas Buettner

- Reliability-centric process
- Reliability estimated from testing results
  - Or from pre-testing characteristics such as “churn”
- Can’t assess reliability from architecture
- Unreliability indicates inadequate architecture
Overall

- Specific architectures have specific advantages and disadvantages (Berg, Boegman presentations)
  - High-level view insufficient for evaluation
- Reference architecture based on UML
  - A number of views of a system
- Would Pope, Sercel, Northrop, Hantos view any of these things as architectures?
- Reliability the most basic ility?
ACE2’s four issues for software architecture (SA)

- SA as basis for understandability
  - Architecture provides common terminology and concepts, basis for relating stakeholder viewpoints

- SA as link between req’s and detailed implementation
  - Evaluate impact of requirements change -- maintainability
  - Provide basis for considering extensibility
  - Assess executability of requirements

- Architecture is “right level” for considering requirements