Lessons Learned and New Directions for Model-Driven Development of Large-Scale Software-Based Systems

Michael J. Marich
Haig F. Krikorian
Architecture Dilemma

• Components of many modern large-scale software systems are distributed, dynamic, and mobile

• [Shaw et al.; Oquendo]: Proper architecture development is key to program development and maintenance

• Architecture polices (control, data, events, behavior over time, etc.) and cross life-cycle development must be reconciled with processes and tools to capture many inter-dependent system policies (correctness, completeness, reducibility, etc.)
Precise Semantics is Required

• [France] A standard modeling notation requires precise semantics for the standard to maintain utility.
• [Clark & Evans]: Precise semantics allows:
  – design consistency
  – design reduction
  – modification correctness
  – translation of requirements to design
• [Bruel & France]: Formalize UML with a semantic base and develop a calculus to support reasoning.
Lessons Learned from Modeling Process

- Expect conflicting project viewpoints
- Establish modeling guidelines
- Gain team member commitment
Lessons Learned from Modeling Tools

- Capture Ground Truth
- Allow for Architectural Refinements
- Be aware of poor transition from architecture to code
- Avoid the UML rut
- Expect an expertise gap
- Manage model access
- Review lessons learned
New Directions – Modeling Process

• CMMI can address many of the process issues
• SDLC is deficient regarding architecture development
  – There is a need to clearly delineate dilemma’s faced by an architect
  – Augment Sub-phases proposed by Garlan & Shaw
New Directions – Modeling Tools

• Software community needs to revisit fundamental issues raised in Lessons Learned
• New semantic and calculus based tools should be transparent to the developer
• New research with π-ADL appears promising
Migration Plans

• Considerable effort is needed for large companies

• Lewin’s 3-stage model:
  – Unfreeze, Move, Refreeze

• Leavitt:
  – Task, technology, people, structure

• Davis’ technology acceptance model:
  – Perceived ease of use and perceived usefulness
References