ULS Ecosystem Design

Research Area: Design

Kevin Sullivan
University of Virginia
Today’s Problem

• Gap between state of art & practice

• Larger than in most other disciplines
Example: Security

• State of practice is still terrible overall

• Many big problems avoidable in principle
Tomorrow’s Problem

• *State of the art itself deeply inadequate*

• “As software’s complexity continues to rise, today’s … problems will become intractable unless fundamental breakthroughs are made in the science and technology of software design and development.” [President’s Council of Advisors on Science and Technology, 07]

• Tomorrow’s problem is here today
Today

• Define and lock **requirements**

• Contract for **development**
  - Partition system & design task: **architecture**
  - Subcontract, implement, and integrate: **code**

• Celebrate **success**
Won’t Work for ULS Systems

• No one adequately understands requirements
• Conditions change (e.g., security/threat environment)
• No one really knows how to build what need to be built
• Complexity and uncertainty pose great challenges
• Once designed, resistant to change (e.g., IPv4 to IPv6)
Major Mismatches
Key Idea

The most critical property of a ULS system is its capacity to adapt to the change dynamics of (including the resolution of risk/uncertainty in) its environment. To be able to assure that given ULS systems have adequate adaptive capacity we need a new discipline of *ecosystem architecture*. Such a discipline will build on but transcend the discipline of software architecture. Economic considerations will play an important role in such a discipline.
Ecosystem Architecture

• Dynamic modeling & monitoring of complex & evolving environments

• Move from an emphasis on architecture of *software* to architecture of *socio-technical ecosystems* of software/system production, operation, use

• Design architecture for **high adaptive capacity** in the given environments

• Coupling of concerns across many levels of socio-technical ecosystem

• Example: security
  – What part(s) of ecosystem will respond to a threat or failure? Autonomic runtime layer? System operators? Software development team? An offensive countermeasures team? Impacts and coordination across multiple levels and administrative domains?
Initial Science Base

- Discipline of software design / architecture
- Structure and economics of modularity in design
- Reactive systems, e.g., for decision support
- Complex adaptive systems, biology
- Network science …
Today

• We’re not even close
• Software architecture today
  – Focus on software artifacts and processes
  – Notations designed accordingly: e.g., UML
  – Not socio-technical ecosystem, environment
  – Box and arrow representations of software and hardware components, interconnections
  – Need to model/structure/analyze and manage dependences among key parameters across whole ecosystems
Today
Tomorrow

- Architecture not about SW and HW components, per se, but about constraints that organize an adaptive optimization process across many levels of a system, including the SW and HW components.

- Fundamental purpose of architecture is to ensure adaptive capacity commensurate with uncertainty & change dynamics of environment.

- Adaptation dynamics in many dimensions, at many levels, at many time-scales.

- Have to design ecosystem, including but not limited to SW/HW, as a key step toward being able to get the SW/HW right.

- Key issues: decentralization & localization, “hiding” of adaptation needs, mechanisms, and dynamics; economic case for doing this.
Structuring Concern Interdependences Across Ecosystem Levels is Critical
Contact

• Me: sullivan@virginia.edu

• ULSSIS Center: http://ulsssis.cs.virginia.edu