From cradle to grave: An architecture substrate for software lifecycles

Nicolas Rouquette
Principal Member of Technical Staff
Jet Propulsion Laboratory
California Institute of Technology
The workflow lifecycle revisited

Is there a consensus that architecture is a pervasive concern?
  - Issue: Architecture as a document vs. an engineering model

Is there a continuity of architecture throughout?
  - Issue: Traceability back and forth among phases

Do we need architecture everywhere?
  - Issue: Representation & semantics of architecture in each phase

Is Architectural change propagation cost-effective?
  - Issue: transforming from one phase to the next (manual, assisted, ...).
The process methodology is flexible...
... as long as State Analysis is applied throughout

Subtle difference between:
- Explicit architecture representation everywhere
- Explicit architecture awareness
- The former is "nice to have"
- The latter is a pragmatic tradeoff for size & fit
State Analysis & SW Architecture

### Code level
- **Type information**
  - Procedural code
  - Functions
  - Classes (OO)

- **Instance information**
  - Variables
  - Events
  - Objects

- **Composition Mechanisms** (how is it built?)
  - Function calls
  - Symbolic references/linkages
  - Shared variables
  - Ad-hoc runtime mechanisms

- **Description Mechanisms** (what's inside?)
  - Ad-hoc runtime mechanisms

### Component Architecture
- **Methods**
- **Interfaces** (sets of methods)
- **Ports** (interface signature)
- **Components** (sets of ports)
- **Connectors** (sets of ports)
- **Mechanical composition**

- **Component instances**
- **Connector instances**
- **Links** (pairs of port bindings)
- **Hierarchical composition**

### System Architecture
- **Types**
  - (from state analysis)
  - State variables
  - Achievers, etc...
  - Domain-specific types
  - Units, etc...

- **Instances**
  - (from state analysis)

- **Subject to lower-level mechanisms**
Architecture Composition in MDS

- xADL extension for component/connector implementation inheritance
- Separation of structure (defined in xADL) & implementation
- Architecture profiling for optimizing transformations of xADL to code
- Packaging of architecture elements into shared objects
- Dynamic registration of architecture elements at shared object init/fini
- Extensible prescription protocols support connector optimizations
- Architecture evolution includes types & instances
  - Type reconfiguration via dynamic object loading/unloading
  - Instance reconfiguration via prescription changes
The architecture workflow

- Key underlying principles
  - Easy to do the right thing
  - Eliminate semantic replication
  - Plan for technology evolution
  - Low buy-in cost of adoption

Model-based transformation, more than just code generation, context sensitive mapping of input/output theories