Current Issues and Future Trends

- Architectural interchange
- Architectural toolkit
- Architectural refinement
- Architectural view integration
- Bringing architectures to the masses

Architectural Interchange

- A number of architecturally-useful tools
  - active specification aids (e.g., agents)
  - analysis
  - refinement
  - simulation
  - code generation
- Some tools are provided for specific ADLs
- Others are provided for specific styles or application domains
- They may be useful more generally
- Two choices
  - implement similar tools across languages/styles/domains
  - interchange architectural descriptions
What Architectural Interchange Entails

- Mapping syntax across architecture specification techniques
  - typically straightforward: components + {connectors} + configurations + Δ

- Mapping semantics across architectural specification techniques
  - difficult task in general
    - CSP <=> FSM
    - posets <=> CSP
    - first-order logic <=> posets

→ ACME addresses the first issue

ACME Goals

- Provide a basis for developing new tools
- Provide a basis for developing new ADLs
- Serve as a representation standardization vehicle
- Understandability by humans

- Simple core
  - fixed ontology
- Everything else pushed to extensions
  - open semantic framework

→ Least common denominator?
ACME Basics

- Structure-centric ADL intended to support architectural interchange
  - can be used to only describe architectures
- Extensible language base for new ADLs
  - systems
  - components/connectors
  - ports/roles
  - attachments
  - representations and rep-maps
- Intended as a platform for ADL-independent tool development
- Caveat: ACME's *core ontology* may not be general enough

ACME Core Structural Features

![Diagram of ACME core structural features](image-url)
ACME Representations

- Provide compositionality support
  - hierarchical abstractions
- Used to represent subarchitectures
- ACME rep-maps
  - define correspondence between (internal) representation and (external) component/connector interface
  - rep-maps may be arbitrarily complex

ACME Properties

- Arbitrary attribute-value pairs
  - intended to represent functional and extra-functional information
  - anything can be made a property
- Can be associated with all major language constructs

Component server = {
  Property max-transactions-per-sec = 5;
  Property max-clients-supported = 200;
  Port rpc-request = {
    Property supports-sync-requests = true;
  }
};
How ACME is Used for Interchange

- Map your ADL's structural features to ACME's

... But

- “Architectural structure forms the skeleton, semantics form the flesh”
- All semantics in ACME are in property lists
  - component computations
  - connector protocols
  - extra-functional properties

→ Who handles their mapping?
  - done on a pairwise basis
  - appears to demand a lot of (hard) work
Simplified Example: From Wright to Rapide

- Claimed benefits
  - easy structural conversion
  - using ACME-based tools for manipulating Wright
    - graphical browsing
    - conversion to WWW documents
    - persistent storage

Architectural Toolkit

- ACME asserts that heterogeneous tools are needed
- What are those tools?
- Two ways of approaching the problem
  - provide the answer up front
    - determine exact set of tools
    - is there a canonical tool suite for architecture-based development?
    - what is its DSSA?
  - allow architects to specify their own tool needs
    - flexible tool integration infrastructure
    - in essence a DSSA
What Tools?

- Architecture editing
- Static and dynamic analysis
- Simulation
- Refinement/Code Generation
- Monitoring execution
- Test case generation
- Static and dynamic evolution
- Architecture viewing
  - e.g., animation
- Configuration management
- Help/Documentation/Annotation/Support

Putting the Tools Together

- Difficult to argue/predict a single, universally useful set of architectural tools
- Flexible integration platform is more useful
- Important principle
  - eat our own dog food
- Issue
  - are arch-based development environments any different really from SDEs?
DRADEL Environment’s Architecture

- Repository
- Internal Consistency Checker
- Parser
- Topological Constraint Checker
- Type Checker
- Code Generator
- User Palette
- Type Mismatch Handler
- Graphics Binding

ArchStudio Environment’s Architecture

- Architecture ADT
- WWW Browser
- Argo
- Arch Shell
- Extension Wizard
Architectural Refinement

- Arriving at an implementation from an architecture
- Fully automated refinement
  - correct implementations by construction
  - does not work in general
  - difficult to establish proper refinement rules
  - mostly method, may involve some theft, no intuition
  - difficult to produce (near) optimal implementations
  - implementation-level errors corrected by regeneration

- Fully human-guided refinement
  - potential for optimal solutions
  - potential to catch errors early
  - combines method, theft, and intuition
  - error prone

→ *Strike a balance between the two*

Architectural View Mismatch Detection and Resolution

- A software system can be viewed from multiple perspectives (i.e., views)
  - data flow
  - control flow
  - application domain objects
  - structural
    - textual vs. graphical
    - conceptual vs. implementation vs. deployment
  - behavioral
  - process
- Information represented in the views is often redundant

→ *Key challenge is to ensure inter-view consistency*
Architectural View Mismatches — Example 1

Possible Mismatch: dependency of Flight to Flight Controller not reflected in lower level view

Architectural View Mismatches — Example 2

Possible Mismatch: creation of Patient not visible in Screen
Architectural View Mismatches — Example 3

View Mismatch Identification Framework

→ Mismatch resolution poses additional problems
Bringing Architectures to the Masses

- Combine the benefits of powerful, specialized notations with those of widely adopted, general notations
  - is UML the answer?

- How should this be done?
  - preserve the benefits of a standard notation
  - augment UML to better serve the needs of architects
    -> UML profiles