Formalization and Investigation of
Software Architecture

February 1997

Capt Mark J. Gerken, Ph.D.
Rome Laboratory/C3CB
525 Brooks Road
Rome NY 13441-4505
gerkenm@rl.af.mil
**ARCHITECTURE & GENERATION**

**PROBLEM**
- Paradigm shift to the development of application families and an increased reliance on component reuse necessitates advanced mechanisms for representing system architecture
  - No commonly accepted definition of architecture beyond notions of components and connectors; emerging ADLs, but terminology is not fixed and competing languages and logics are used
  - Difficulty predicting/analyzing functional/extra-functional characteristics of integrated components
  - Architectural archeology (e.g., recovery of architectural information from legacy systems)

**APPROACH**
- Develop common architecture specification notions/languages to facilitate multiple views through interrelated ADLs
- Develop integrated architectural analysis, design, and measurement capabilities to assist in understanding functional and extra-functional characteristics
- Automate composition/synthesis of application implementations from architectural specifications
- Preserve all aspects of system being developed including behavioral and architectural specifications to enable subsequent examination, modification, and/or transformation

**PAYOFF**
- Supports static and dynamic analysis of system properties prior to system implementation
- Architecture is formally specified and used to generate/develop/evolve source code
  - Architecturally consistent implementation
  - Cost-effective
  - Easily evolvable
## ARCHITECTURE & GENERATION

### CAPABILITIES BREAKOUT

#### Software Generators

- Given architectural and behavioral specifications, reusable artifacts, and knowledge of the problem domain, produce executable systems consistent with the specifications

  • Understand the relationships between the compositional and the transformational approaches to software generation
  
  • Develop techniques for representing and reusing software artifacts
  
  • Generalize generator technology to remove dependence on implicit or hard-coded target architectures
  
  • Develop a common semantic framework for architecture-based software generation
  
  • Develop technologies to help build generators

#### Constraints

- Tighten the design space for a system by constraining the run-time behavior of a system or by constraining the development process

  • Develop techniques for representing and reasoning about constraints
  
  • Develop techniques for responding to observed, pending, or potential constraint violations
  
    - Monitoring and reporting: report violation of run-time constraints
  
    - Correction: after constraint violation detected, system will fall back into an acceptable state
  
    - Prevention: monitor system state for conditions that may lead to a constraint violation and take corrective action to avoid the violation
  
    - Preclusion: guarantee constraint satisfaction through construction techniques
- Develop a common semantic framework for ADLs to support tool sharing and information exchange
  - Develop a common ADL capable of representing the key structures of existing ADLs, and define transformations into and out of this common ADL
  - Determine the structural information ADLs must be able to represent
  - Determine the types of analysis ADLs must support
  - Investigate approaches for representing and using architectural styles

- Develop technologies to support architectural analysis and refinement
  - Model architectures in terms of event sequences
  - Static and dynamic analysis over multiple ADLs
  - Correctness evaluation between levels of description
  - Develop modeling and analysis tools and techniques supporting families and styles of architectures, including support for architectural mismatch

- Develop tools and techniques to support the formal investigation of software architecture properties
  - Evaluate alternative architectures in the context of expected environment
  - Develop evaluation techniques, such as data flow, graph reachability, and integer programming
GTN is a “Federal Express” system for planning and tracking the global transportation of military
cargo, personnel, and patients in support of various missions using air, sea, and land-based assets

- Architecture: Distributed heterogeneous, with aspects of MIS and command and control
- May be some link to information management/rationale capture (TBD)

Global Transportation Network (GTN): Lockheed Martin

- GTN is a “Federal Express” system for planning and tracking the global transportation of military
cargo, personnel, and patients in support of various missions using air, sea, and land-based assets
- Architecture: Distributed heterogeneous, with aspects of MIS and command and control
- May be some link to information management/rationale capture (TBD)

High Level Architecture (HLA): CMU and Stanford

- HLA is a standard developed for the distributed simulation domain, where federates can join and
leave a simulation at unpredictable times
- HLA standard has been modeled using a couple of event-based ADLs. Result: errors in the specification were detected!
A Categorical Framework for Specification Refinement

Define an environment for constructing specifications for complex systems
- First-order specifications used to define data types and operations
- Process-based specifications used to define sequencing, communication, and synchronization
- Sponsor: AFOSR
SOURCES FOR ADDITIONAL INFORMATION

DARPA

Dr. John Salasin
jsalasin@darpa.mil

Dr. Howard Shrobe
hshrobe@darpa.mil

Rome Laboratory
http://www.se.rl.af.mil:8001/edcs/

Frank S. LaMonica
lamonicaf@rl.af.mil

Douglas A. White
whited@rl.af.mil

Software Engineering Inst.
http://www.sei.cmu.edu/~edcs/

Doug Waugh
dww@sei.cmu.edu

David Carney
djc@sei.cmu.edu

Note: Web Sites Currently Under Construction