Research Directions in Software Architecture

February 1998

Capt Mark J. Gerken, Ph.D.
AFRL/IFTD
525 Brooks Road
Rome NY 13441-4505
gerkenm@rl.af.mil
Overview

• Vision

• General Research Areas:
  – Architecture Representation (styles)
  – Transforming and Communicating Architectures
  – Architecture-based Analysis
  – Architecture-based Generation

• Further Information
Systems Are:
- Specified
- Designed
- Analyzed
- Built
- Tested

Through Architecture
- Provide Manipulation & Analysis Tools
- Make Architecture Explicit & Formal

Architecture Vision

Architecture-Based Development And Evolution

Evolved
Architecture Representation

- **Software Architecture involves:**
  - descriptions of elements from which systems are built,
  - interactions among those elements,
  - patterns that guide their composition, and
  - constraints on those patterns.

- **Goals:**
  - To provide a scientific and engineering basis for design, analysis, and composition of flexible systems from complex building blocks;
  - To provide languages, tools, environments, and techniques to support the above goal
Architecture Representation

• Progress:
  – Unicon: components are loci of computation and state, connectors are loci of relations between components
  – Wright: defines communication between components
  – Aesop: system for developing style-specific architectural development environments. Aesop was used by Lockheed-Martin to provide a front-end design environment for Global Transportation Network (GTN)
  – Jakarta: generator environment for composing systems
  – Specware: Architecture as a diagram of formal specifications

• Future Work:
  – Rehost the UniCon toolset to support Wintel platforms
  – Jakarta applied to radio domain
  – Specware applied to transportation domain (Planware)
Transforming and Communicating Architectures

- Supports common static analysis services
- Provides tool access through architecture description language (ADL) translation
- Supports architectural interchange
ACME Specification

System simple_cs = {
    Component client = {  Port send-request;
                           Properties {
                                request-rate : float = 17.0;
                                source-code : external-file = "client.c"  }
    }
    Component server = {  Port receive-request;
                           Properties {
                                idempotence : boolean = true;
                                max-clients : integer = 1;
                                source-code : external-file = "server.c"  }
    }
    Connector rpc = { Role caller;
                     Role callee;
                     Properties { synchronous : boolean = true;
                                  max-roles : integer = 2;
                                  protocol : Wright = "..."  }
    }
    Attachments {  client.send-request to rpc.caller ;
                    server.receive-request to rpc.callee }
}
Architecture-Based Analysis

• Formality supports analysis
  – Static checks, e.g.
    » Ambiguities
    » Incompleteness
    » Wrong Directionality
  – Model Checking
    » Insufficient Preconditions
    » Faulty Control Model
    » Latent Deadlocks
  – Simulation-based Testing
    » Event Order Anomalies
    » Causality Anomalies

Example: DMSO
Simulation Framework (HLA)

Wright (CMU)
• Distributed Startup
• Paused on Join
• Intransit msgs after Resign

Rapide (Stanford)
• Run Time Interface lost event order
• Orphaned attrs after resign
MIC APPROACH IN THE MGA:

- **MODEL INTEGRATION:** Multiple, “overlapping” views with maintained constraints

- **MODEL TRANSFORMATION:** Model Interpreters translate domain models into analysis and executable models

- **APPLICATION INTEGRATION:** run-time support for executable models
SOURCES OF ADDITIONAL INFORMATION

DARPA

AFRL-Rome

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

* Start Here

Dr. John Salasin
jsalasin@darpa.mil

Frank S. LaMonica
lamonicaf@rl.af.mil

Douglas A. White
whited@rl.af.mil