“Applying Architecture Modeling Methodology to Enterprise Software Domains”
Applying Architecture Modeling Methodology to Enterprise Software Domains

Presented at the 23rd Systems and Software Technology Conference (SSTC), 16-19 May 2011, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License

Abstract

Security classification of:
- Report: unclassified
- Abstract: unclassified
- This Page: unclassified

Limitation of abstract: Same as report (SAR)

Number of pages: 25
About Dr. Rivera

- President of Rivera Group
- **Specialization**: Enterprise Software Development, Research & Development, Business Process Reengineering (BPR)
- **R&D Projects**: Modeling & Simulation, Natural Language Processing (NLP) of Open Source data for the Intelligence Community

- SBA 8(a) Small Disadvantaged Business
- Service–Disabled Veteran Owned Small Business
- HubZone Certified Minority–Owned Business
Journal Publications


Predict System Failures
Before You Build

Question: “I have a great idea, but what happens if I change my system architecture?”

Eagle6 is a modeling and simulation tool that is capable of predicting system behavior.

- Ensure Changes to System Architecture Do Not Cause Unintended System Behavior
- Validates Architectural Designs That Mix Legacy Systems and New Technologies
- Evaluates Business Process Reengineering Designs (Lean Six Sigma)

Answer: Test BEFORE You Invest!
Problem: What could happen if I upgrade/change this system?

Coming Up: How we solve this problem...
Problem Statement

“System Architecture Designs May Contain Unintended and/or Unknown System Behavior.”
Issues with Requirements

Typical difficulties

- Users initial concept of system is nebulous
- Users description of system is incomplete and inconsistent
- Users (usually) don't understand what they really need
- Interpreting users description of problem is error-prone
- Perception of system changes during analysis, requires reworking
- Different users will view the system differently

**Question:** How do we know if the system architecture represents all system requirements?  
**Answer:** Model Checking
Produce Better Software

- Major goal of software engineers
  - Develop reliable systems
- Formal Methods
  - Mathematical languages, techniques and tools
  - Used to specify and verify systems
  - Goal: Help engineers construct more reliable systems
- A mean to examine the entire state space of a design (whether hardware or software)
  - Establish a correctness or safety property that is true for all possible inputs
Problems with Formal Methods

- Past years of the formal methods
  - Obscure notation
  - Non-scalable techniques
  - Inadequate tool support
  - Hard to use tools
  - Very few case studies
  - Not convincing for practitioners

- Bottom Line: It’s not easy.
Unknown System Behavior

Key Goal: Find Unknown or Unintended System Behavior
Model Checking

Pros

- Great for system safety testing.
  - Medical Systems
  - Weapon Systems
- Great for finding unknown system behavior and/or architectural design flaws (assertion checking)

Cons

- Modeling languages are very complex and require domain expertise
- Models require a very long time to develop
- Modifying models is not easy, making reuse very difficult
Test Using One Model

Using a single model allows for system constraints to remain resident throughout all stages of the system lifecycle.
Summary of Formal Methods

- Formal methods can be applied at various points through the development process
  - Specification
  - Verification
- **Specification**: Give a description of the system to be developed, and its properties
- **Verification**: Prove or disprove the correctness of a system with respect to the formal specification or property
Eagle6 Demonstration (www.Eagle6.com)
Problem: What could happen if I upgrade/change this system?

Answer: Model the solution and test, test, test!
Using a single model allows for system constraints to remain resident throughout all stages of the system lifecycle.
Model the Enterprise Without a Single Line of Code

Tree structure showing event parent/child relationship

Event container properties

Graphical representation of model

Event attributes
Eagle6 Model in Text Form
Eagle6 Graphical Editor

Graphical editor is easy to use by most end users.

Refine your model when you learn more about your architecture.
Test SoS Assertions
1) **Event Count** – Check for the existence of a specific system state

2) **Sequence of Events** – can a series of events happen?

3) **Simultaneous Events** – Can a combination of events happen?
Run All System Tests with One Model in Seconds

Execute all system tests to determine potential architecture flaws

Model Failure? Click on the link to view WHY the model fails
Scenario Viewer

Graphical view of the exact sequence of events that violated the assertion
Eagle6 Summary

1. Modeling and Simulation software tool that is used to dynamically model any type of complex enterprise system to identify **risks** in system architecture.

2. Checks all possible system states within the model scope.

3. Capable of executing all types of system tests within a single model.

4. Modeling interface that allows a user to write models without having to learn a modeling language (tool is designed for the average user)
Questions?

Dr. Joey Rivera
jrivera@riverainc.com

Phil Lushin
plushin@riverainc.com

812-246-4055