



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

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 16 MAY 2011		2. REPORT TYPE		3. DATES COVERED 00-00-2011 to 00-00-2011	
4. TITLE AND SUBTITLE Applying Architecture Modeling Methodology to Enterprise Software Domains				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Rivera Group,555 Maple Ave,Sellersburg,IN				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES 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					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 25	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

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

- Rivera, J., Auguston, M., Finkbine, R. (2011) Modeling Methodology for Validation and Verification of System Architecture Designs, 23rd Annual Systems & Software Technology Conference (SSTC 2011), Salt Lake City, Utah.
- Rivera, J. (2010). Applying Architecture Modeling Methodology to the Naval Gunship Software Safety Domain. ACM/IEEE 13th International Conference on Model Driven Engineering Language and Systems. Oslo, Norway.
- Rivera, J., & Luqi. (2010). Requirements Framework for the Software System Safety Technical Review Panel Technical Review Package. Monterey, CA: Naval Gunnery Project Office PEO WS3C.
- Rivera, J., Luqi, and Berzins, V. (2009) Effective Programmatic Software Safety Strategy for US Navy Gun System Acquisition Programs, 6th Annual Acquisition Research Symposium of the Naval Postgraduate School:, Vol. 2, Monterey, CA.

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

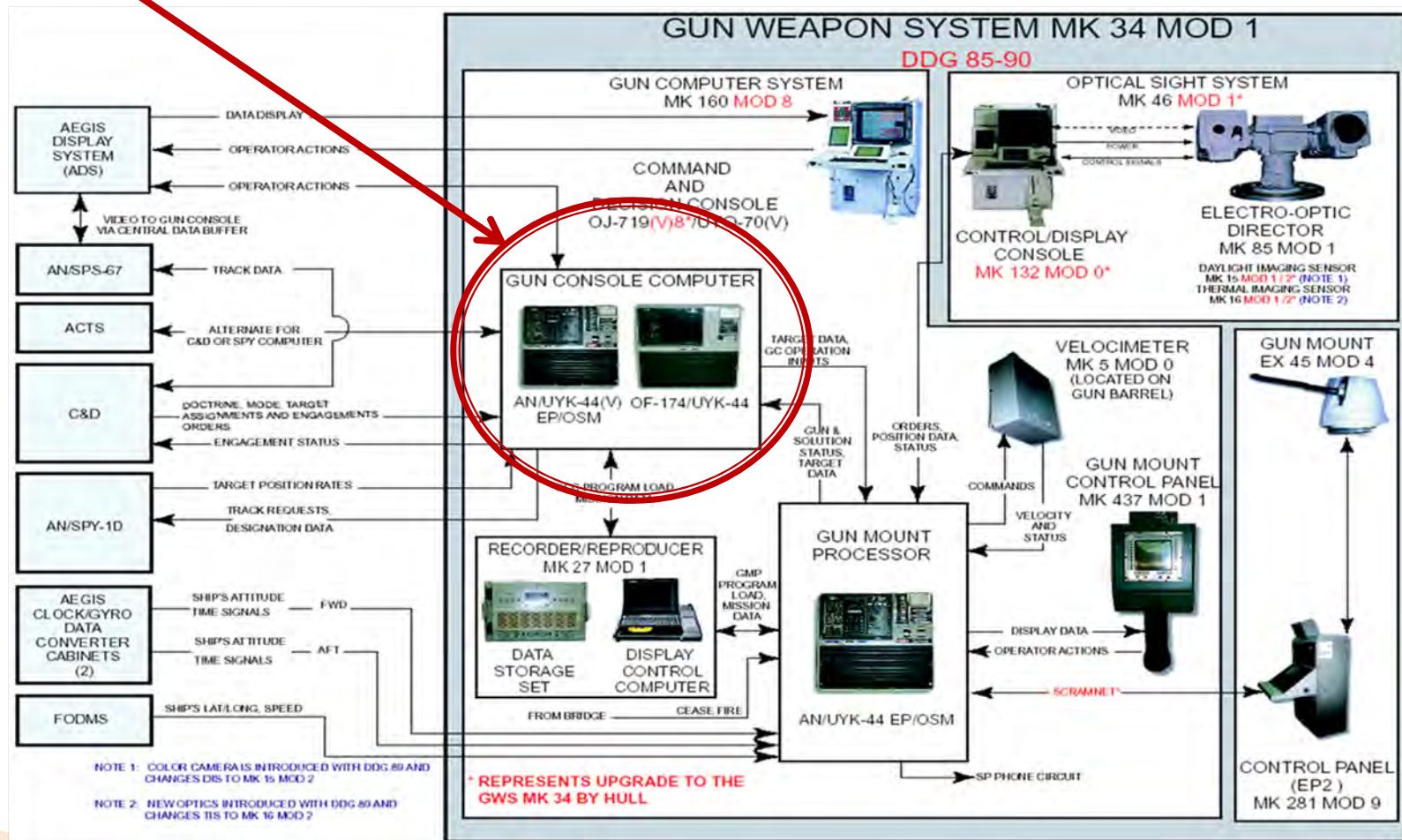
Validates Architectural Designs That Mix Legacy Systems and New Technologies

Evaluates Business Process Reengineering Designs (Lean Six Sigma)

Answer: Test **BEFORE** You Invest!

System of System Dependencies

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

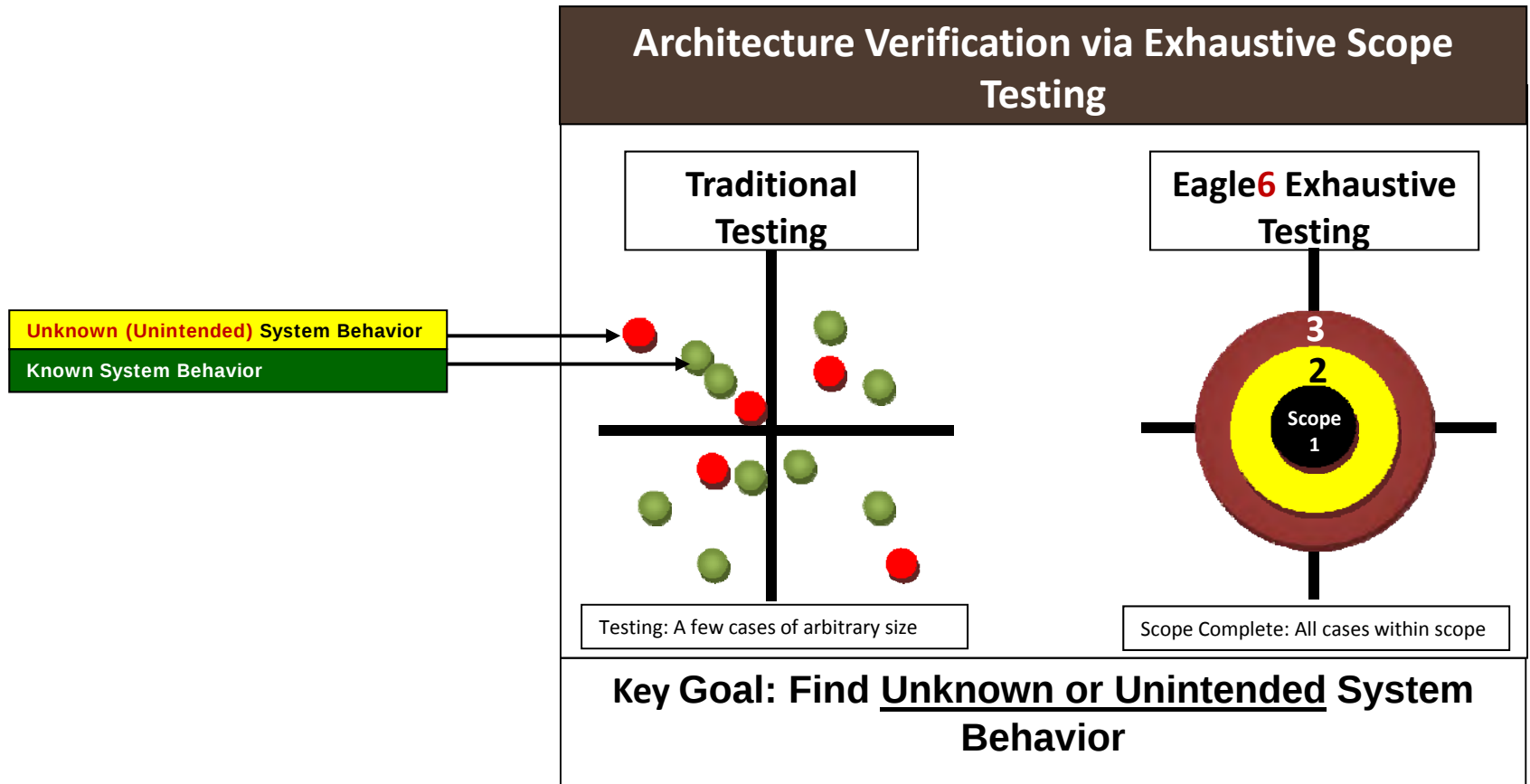
- ▶ Major goal of software engineers
 - Develop reliable systems
- ▶ Formal Methods
 - Mathematical languages, techniques and tools
 - Used to specify and verify systems
 - Goal: Help engineers construct more reliable systems
- ▶ A mean to examine the entire state space of a design (whether hardware 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



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

Requirements Testing

Conformance Testing

Functional Testing

Integration Testing

Black Box Testing

Performance Testing

Regression Testing

System Testing

Unit Testing

Eagle6
Model

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

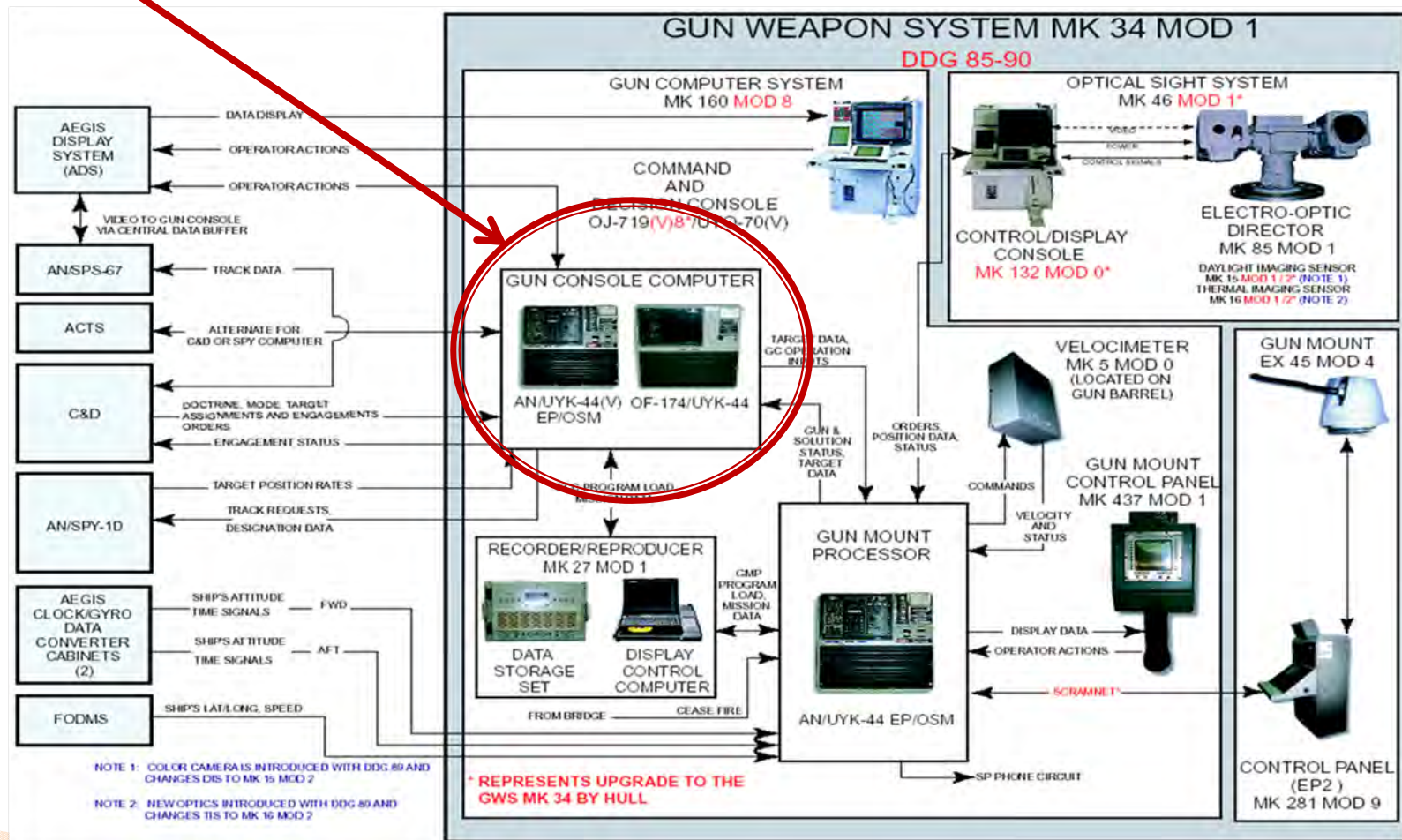
Summary of Formal Methods

- ▶ 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)

Old SoS Problem, New Solution

Problem: What could happen if I upgrade/change this system?



Answer: Model the solution and test, test, test!

Reminder: Test Using One Model

Requirements Testing

Conformance Testing

Functional Testing

Integration Testing

Black Box Testing

Performance Testing

Regression Testing

System Testing

Unit Testing

Eagle6
Model

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

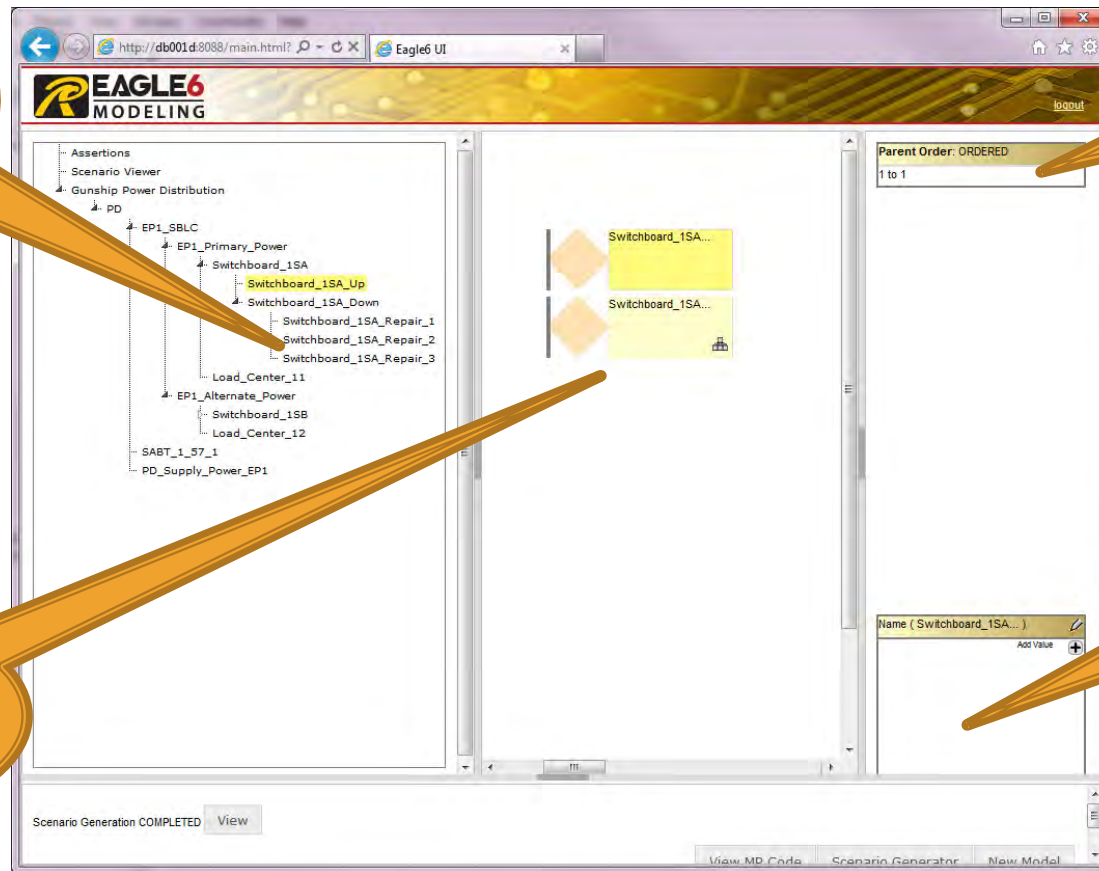
Model the Enterprise Without a Single Line of Code

Tree structure showing event parent/child relationship

Event container properties

Graphical representation of model

Event attributes



Eagle6 Model in Text Form

Eagle6 UI - Windows Internet Explorer

http://db001d:8088/main.html?model_id=16

File Edit View Favorites Tools Help

Google Search More»

US Airways | Air... Eagle6 UI Military Symbols | US Airways | Air...

EAGLE6 MODELING

Scenario Viewer

- Naval Gunship
 - R2D_activity
 - CD_activity
 - GCC_activity
 - R3D_activity
 - GCC2_activity
 - GMP_activity
 - GMCP_activity
 - GMP2_activity
 - CDC_activity
 - EOD_activity
 - GM_activity

Preview Code

```
ROOT R2D_activity: { * <1> R2D_displayNewTarget * };
ROOT CD_activity: { * <1> CD_spotNewTarget * };
ROOT GCC_activity: { * <0-1/0.2,0.8> GCC_setTarget * };
ROOT R3D_activity: { * <0-1/0.28,0.72> R3D_setTarget * };
ROOT GCC2_activity: { * <0-1/0.44128,0.55872> GCC_openFire * };
ROOT GMP_activity: { * <0-1/0.452454,0.547546> GMP_answerRequest_GCC_openFire * };
ROOT GMCP_activity: { * <0-1/0.45793,0.54207> GMCP_answerFireRequest * };
ROOT GMP2_activity: { * <0-1/0.463351,0.536649> GMP_answerRequest_GMCP_ossData * };
ROOT CDC_activity: { * <0-1/0.490183,0.509817> CDC_answerRequest_GMP_ossData * };
ROOT EOD_activity: { * <0-1/0.495281,0.504719> EOD_answerRequest_CDC_ossData * };
ROOT GM_activity: { * <0-1/0.505013,0.494987> GM_answer_GMCP_openFireCommand * };

E1: ( GCC_targetNotSet CD_targetLost );
B1: ( CD_request_GCC_openFire CD_wait_GCC_openFire ( GCC_openFireFailed |<0.25249031177832> targetMissed |<0.58914406084842> targetHit ) );
D1: ( GCC_targetSet CD_followTarget ( CD_abortTarget |<0.8> B1 ) );
G1: ( CD_request_GCC_setTarget CD_wait_GCC_setTarget ( E1 |<0.873> D1 ) );
CD_spotNewTarget: ( R2D_displayNewTarget ( CD_ignoreTarget |<0.8> G1 ) );
B2: ( R3D_targetNotSet GCC_targetNotSet );
A2: ( R3D_targetSet GCC_targetSet );
D2: ( GCC_request_R3D_setTarget GCC_wait_R3D_setTarget ( B2 |<0.97> A2 ) );
GCC_setTarget: ( CD_request_GCC_setTarget ( GCC_targetNotSet |<0.9> D2 ) );
R3D_setTarget: ( GCC_request_R3D_setTarget ( R3D_targetNotSet |<0.97> R3D_targetSet ) );
A3: ( GMP_openFireFailed GCC_openFireFailed );
B3: ( GCC_request_GMP_openFire GCC_wait_GMP_openFire ( A3 |<0.257643175284> targetMissed | targetHit ) );
GCC_openFire: ( CD_request_GCC_openFire ( GCC_openFireFailed |<0.98> B3 ) );
A4: ( GMCP_openFireFailed GMP_openFireFailed );
B4: ( GMCP_displayOpenFireRequest ( A4 |<0.6072398071> targetHit |<0.2602456316> targetMissed ) );
GMP_answerRequest_GCC_openFire: ( GCC_request_GMP_openFire ( GMP_openFireFailed |<0.99> B4 ) );
D5: ( GMCP_failReceiving_GMP_ossData GMCP_openFireFailed );
A5: ( GM_openFireFailed GMCP_openFireFailed );
B5: ( GMCP_send_GM_openFireCommand GMCP_wait_GM_openFireCommand ( A5 |<0.665> targetHit |<0.285> targetMissed ) );
C5: ( GMCP_receive_GMP_ossData B5 );
E5: ( GMCP_request_GMP_ossData GMCP_wait_GMP_ossData ( D5 |<0.9223662294> C5 ) );
GMCP_answerFireRequest: ( GMCP_displayOpenFireRequest ( GMCP_openFireFailed |<0.99> E5 ) );
B6: ( GMP_failReceiving_CDC_ossData GMCP_failReceiving_GMP_ossData );
A6: ( GMP_receive_CDC_ossData GMCP_receive_GMP_ossData );
C6: ( GMP_request_CDC_ossData GMP_wait_CDC_ossData ( B6 |<0.95089302> A6 ) );
GMP_answerRequest_GMCP_ossData: ( GMCP_request_GMP_ossData ( GMCP_failReceiving_GMP_ossData |<0.95> C6 ) );
B7: ( CDC_failReceiving_EOD_ossData GMP_failReceiving_CDC_ossData );
A7: ( CDC_receive_EOD_ossData GMP_receive_CDC_ossData );
C7: ( CDC_request_EOD_ossData CDC_wait_EOD_ossData ( B7 |<0.960498> A7 ) );
```

Scenario Generation COMPLETED View

Local intranet | Protected Mode: Off

Eagle6 Graphical Editor

The screenshot displays the Eagle6 Graphical Editor interface. The top window title is "US Airways | Airl... Eagle6 UI". The left sidebar contains a project tree for "Naval Gunship" with the following structure:

- Assertions
- Scenario Viewer
- Naval Gunship
 - R2D_activity
 - R2D_displayNewTarget
 - CD_activity
 - CD_spotNewTarget
 - R2D_displayNewTarget
 - CD_ignoreTarget
 - G1
 - GCC_activity
 - R3D_activity
 - GCC2_activity
 - GMP_activity
 - GMCP_activity
 - GMP2_activity
 - CDC_activity
 - EOD_activity
 - GM_activity

The main workspace shows a graphical model diagram. A yellow box labeled "R2D_displayNewT..." has an arrow pointing to a yellow box labeled "CD_ignoreTarget". The "CD_ignoreTarget" box has a "Refine Edit" button. Below this, a yellow box labeled "G1" is connected to the "CD_ignoreTarget" box via a vertical line with a "0.8" value. A small icon is visible in the bottom right corner of the "G1" box.

Graphical editor is easy to use by most end users.

Refine your model when you learn more about your architecture.

Test SoS Assertions

Requirements Testing

Conformance Testing

Functional Testing

Integration Testing

Black Box Testing

Performance Testing

Regression Testing

System Testing

Unit Testing

The screenshot displays the EAGLE6 MODELING software interface. The left pane shows a hierarchical tree of assertions for a 'Naval Gunship' scenario, including activities like R2D_activity, CD_activity, and GCC_activity. The main pane is titled 'Naval Gunship Test Scenarios' and features a table of test events. A dropdown menu is open for 'Test Category', listing various testing types. Below the table, there is a 'Sequenced Events' section with another table. The interface also includes a top toolbar with icons for US Airways, Airline, Eagle6 UI, and Military Symbols.

Naval Gunship Test Scenarios

Test Type: Show All Test Category: SHOW ALL

Category:

Test	Event	Occurs	N times	Min Scope
☐	GMP_openFireFailed	>	90	1
☐	GM_launchMissile	>=	1	1
☐	GMP_openFireFailed	=	0	1
☐	A5	<	4	2
☐	targetMissed	>	2	1
☐	GCC_targetSet	>	0	1

Test All Add New Test

Sequenced Events

Category:

Test	Attribute	Is	Sum	Min Scope	Result	Failed Scenarios
☐	Max_Watts	=	2	5		

Key: Tests via Queries

- 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

The screenshot displays the EAGLE6 MODELING software interface. On the left, a tree view shows the model structure under 'Assertions' and 'Scenario Viewer', including 'Gunship Power Distribution' and 'PD'. The main panel, titled 'Model 14 Test Scenarios', shows test results for 'Category: SHOW ALL'. It includes three tables: 'Event Counts', 'Sequenced Events', and 'Simultaneous Events'. The 'Event Counts' table shows a failure for 'Load_Center_12' with a yellow result. The 'Sequenced Events' table shows a failure for 'Repair_Time_Minutes' with a red result. The 'Simultaneous Events' table shows a success for 'Repair_Time_Minutes' with a green result. Two orange callout boxes provide instructions: one points to the 'Load_Center_12' failure, stating 'Execute all system tests to determine potential architecture flaws', and the other points to the 'Repair_Time_Minutes' failure, stating 'Model Failure? Click on the link to view WHY the model fails'.

Model 14 Test Scenarios

Test Type: Show All Test Category: SHOW ALL

Event Counts

Test	Event	Occurs	N times	Min Scope	Result	Failed Scenarios
☒	Switchboard_1SA_Down	>	1	1	Red	5 6 7 8
☒	EP1_SBL_C	>=	1	1	Green	
☒	Load_Center_12	>	3	6	Yellow	Model is not valid for given scope!

[Test All](#) [Add New Test](#)

Sequenced Events

Test	Attribute	Is	Sum	Min Scope	Result	Failed Scenarios
☒	Repair_Time_Minutes	>	1	1	Green	
☒	Repair_Time_Minutes	=	30	1	Red	4 8 10 11
☒	Repair_Time_Minutes	=	2	6	Yellow	Model is not valid for given scope!

[Test All](#) [Add New Test](#)

Simultaneous Events

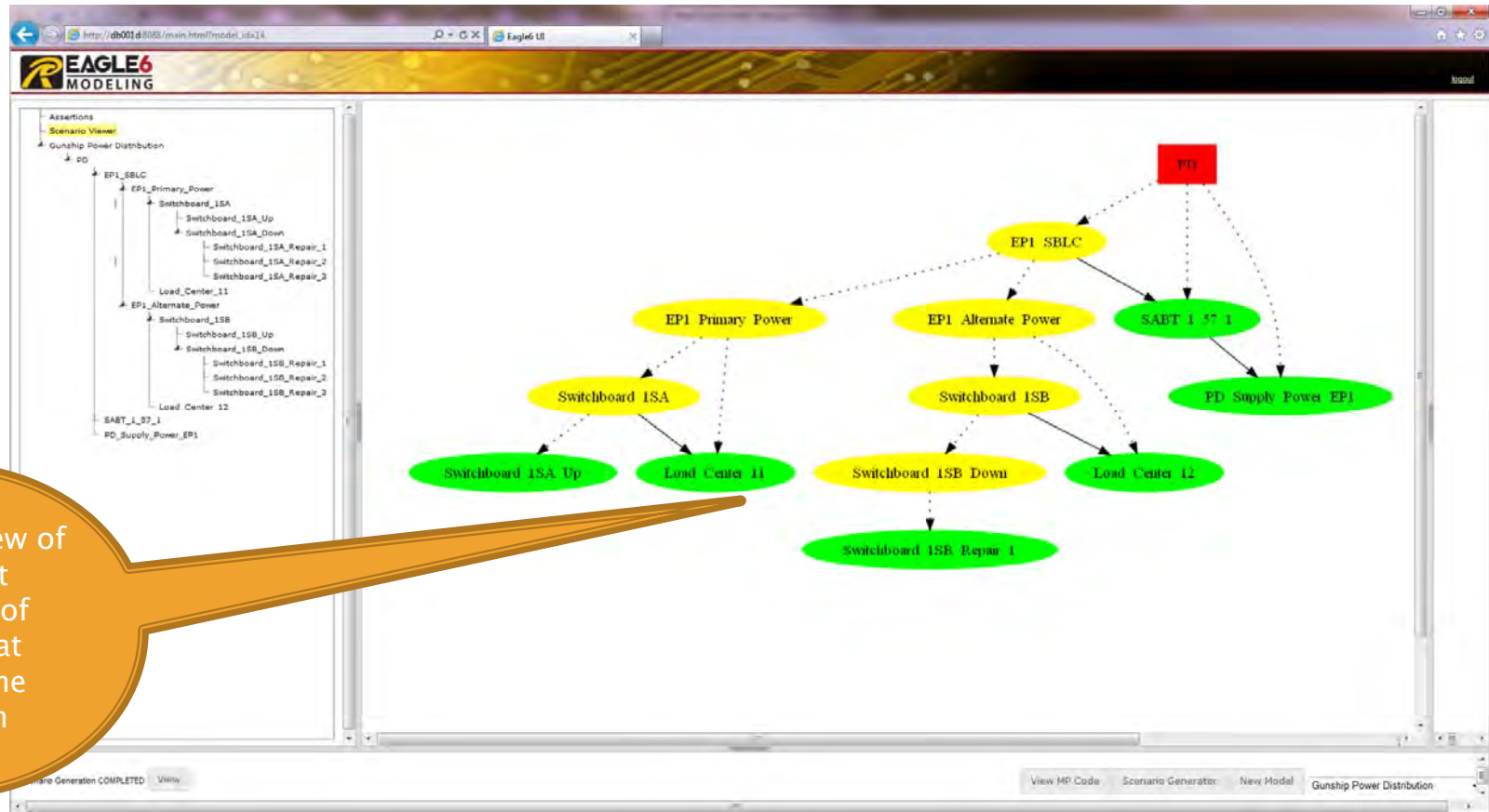
Test	Attribute	Is	Sum	Min Scope	Result	Failed Scenarios
☒	Repair_Time_Minutes	=	1	2	Green	

[Test All](#) [Add New Test](#)

Scenario Generation COMPLETED [View](#)

[View MP Code](#) [Scenario Generator](#) [New Model](#) Gunship Power Distribution

Scenario Viewer



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

