



Bridging the Gap Between Model-Based Development and Model Checking

AFRL Safe & Secure Systems & Software Symposium

Dr. Steven P. Miller

**Rockwell
Collins**

Acknowledgements

- **NASA Langley Research Center (Ricky Butler)**
- **Air Force Research Labs**
- **University of Minnesota (Dr. Mats P. E. Heimdahl)**
- **Lockheed Martin**
- **Dr. Mike Whalen**
- **Dr. Darren Cofer**

Presentation Overview



Who Are We?

What Problem are We Solving?

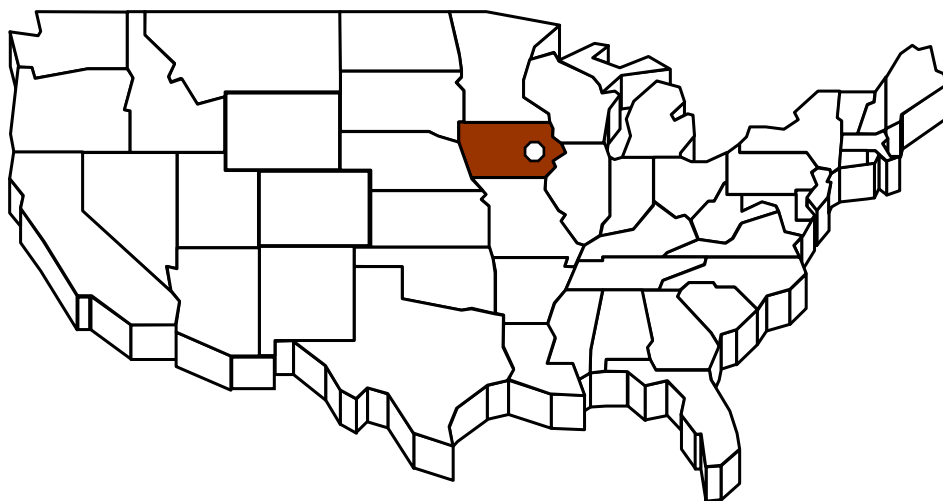
Overview of Our Approach

Case Studies

Challenges and Future Directions

Rockwell Collins

- **Headquartered in Cedar Rapids, Iowa**
- **20,000 Employees Worldwide**
- **2008 Sales of \$4.77 Billion**



Domestic

California

Carlsbad
Cypress
Irvine
Los Angeles
Pomona
Poway
San Francisco
San Jose
Tustin

Florida

Melbourne
Miami
Orlando

Georgia

Atlanta
Warner Robins

Hawaii

Honolulu

Illinois

Chicago

Iowa

Bellevue
Coralville
Decorah
Manchester

Kansas

Wichita

Maryland

White Marsh

Massachusetts

Boston

Michigan

Ann Arbor
Detroit

Minnesota

Minneapolis

Missouri

Kansas City
St. Louis

New York

New York

North Carolina

Charlotte
Raleigh

Oklahoma

Midwest City

Tulsa

Oregon

Portland

Pennsylvania

Philadelphia
Pittsburgh

Texas

Dallas
Fort Worth
Richardson

Utah

Salt Lake City

Virginia

Sterling
Warrenton

Washington

Kirkland
Renton
Seattle

Washington, DC

International

Africa

Johannesburg, South Africa

Asia

Bangkok, Thailand
Beijing, China
Hong Kong
Hyderabad, India
Kuala Lumpur, Malaysia
Manila, Philippines
Moscow, Russia
Osaka, Japan
Shanghai, China
Singapore
Tokyo, Japan

Australia

Auckland, New Zealand
Brisbane, Australia
Melbourne, Australia
Sydney, Australia

Canada

Montreal
Ottawa

Europe

Amsterdam, Netherlands
Frankfurt, Germany
Heidelberg, Germany
London, England
Lyon, France
Manchester, England
Paris, France
Reading, England
Rome, Italy
Toulouse, France

Mexico

Mexicali

South America

Santiago, Chile
Sao Jose dos Campos, Brazil
Sao Paulo, Brazil

Rockwell Collins' core business is based on the delivery of *High Assurance Systems*

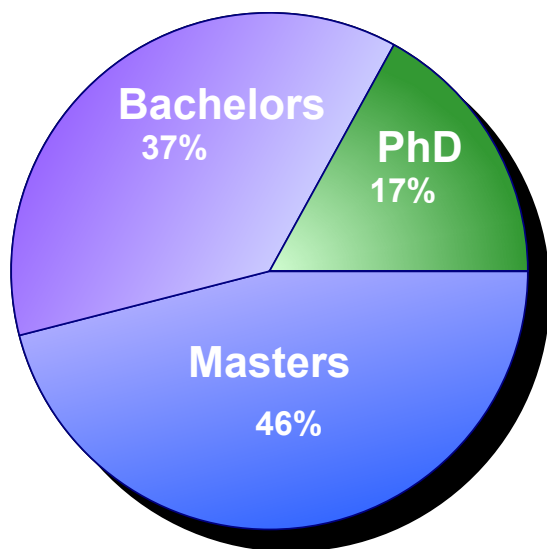
- **Commercial/Military Avionics Systems**
- **Communications**
- **Navigation & Landing Systems**
- **Flight Control**
- **Displays**



“Working together creating the most trusted source of communication and aviation electronic solutions”

Advanced Technology Center

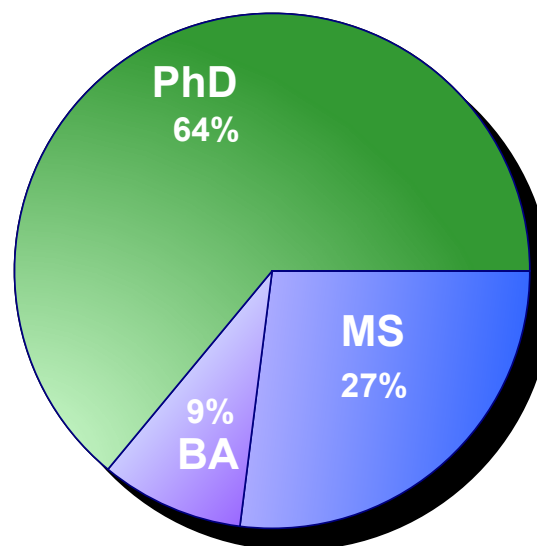
Identify, acquire, develop and transition value-driven technologies to support the continued growth of Rockwell Collins.



Technologists: 173
Administrators: 10
Technicians: 31

Automated Analysis Section

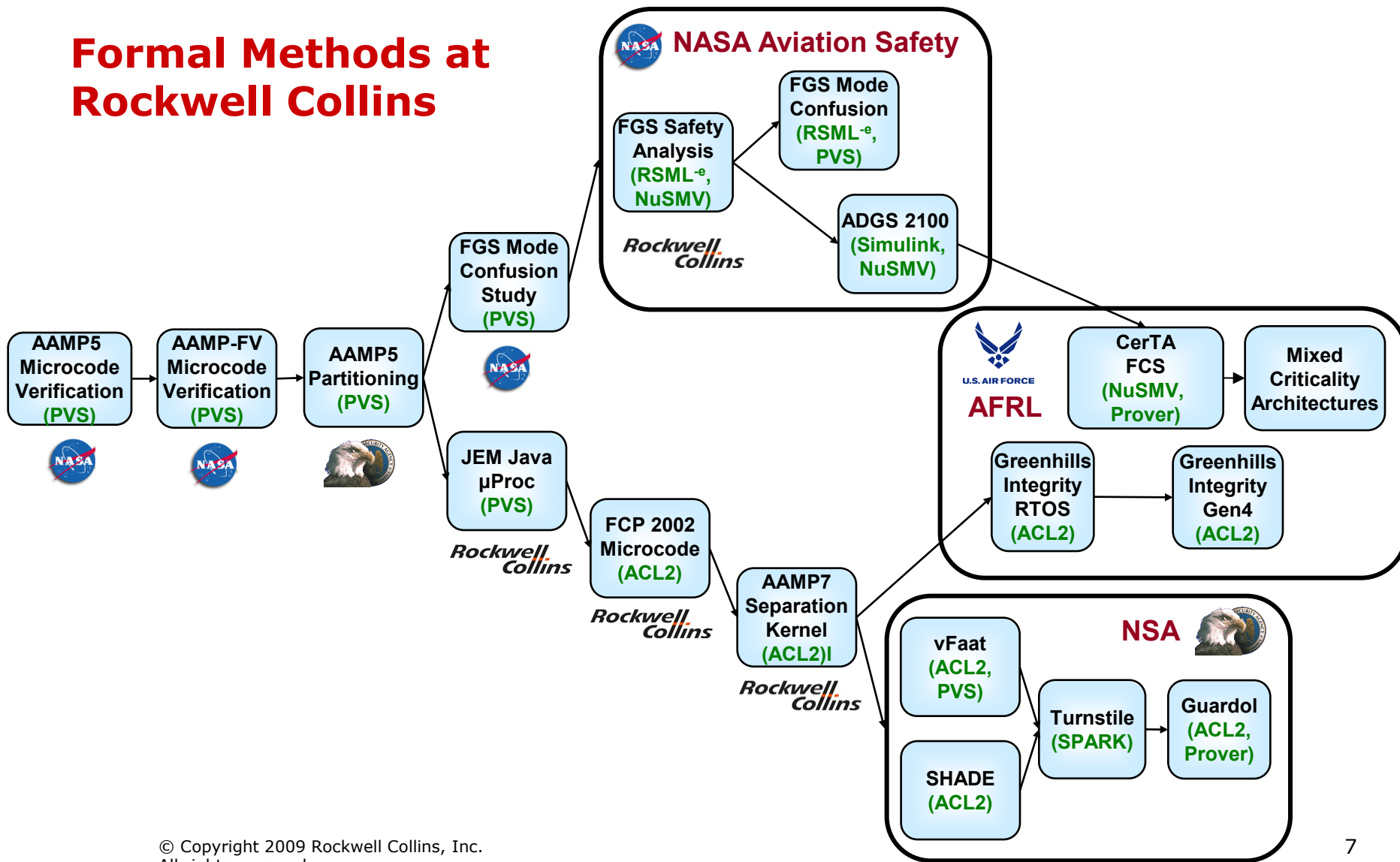
Technologists: 10
Administrators: 1



Applies mathematical tools and reasoning to the production of high assurance systems.

1992 1994 1996 1998 2000 2002 2004 2006 2008 2010

Formal Methods at Rockwell Collins



Presentation Overview

Who Are We?



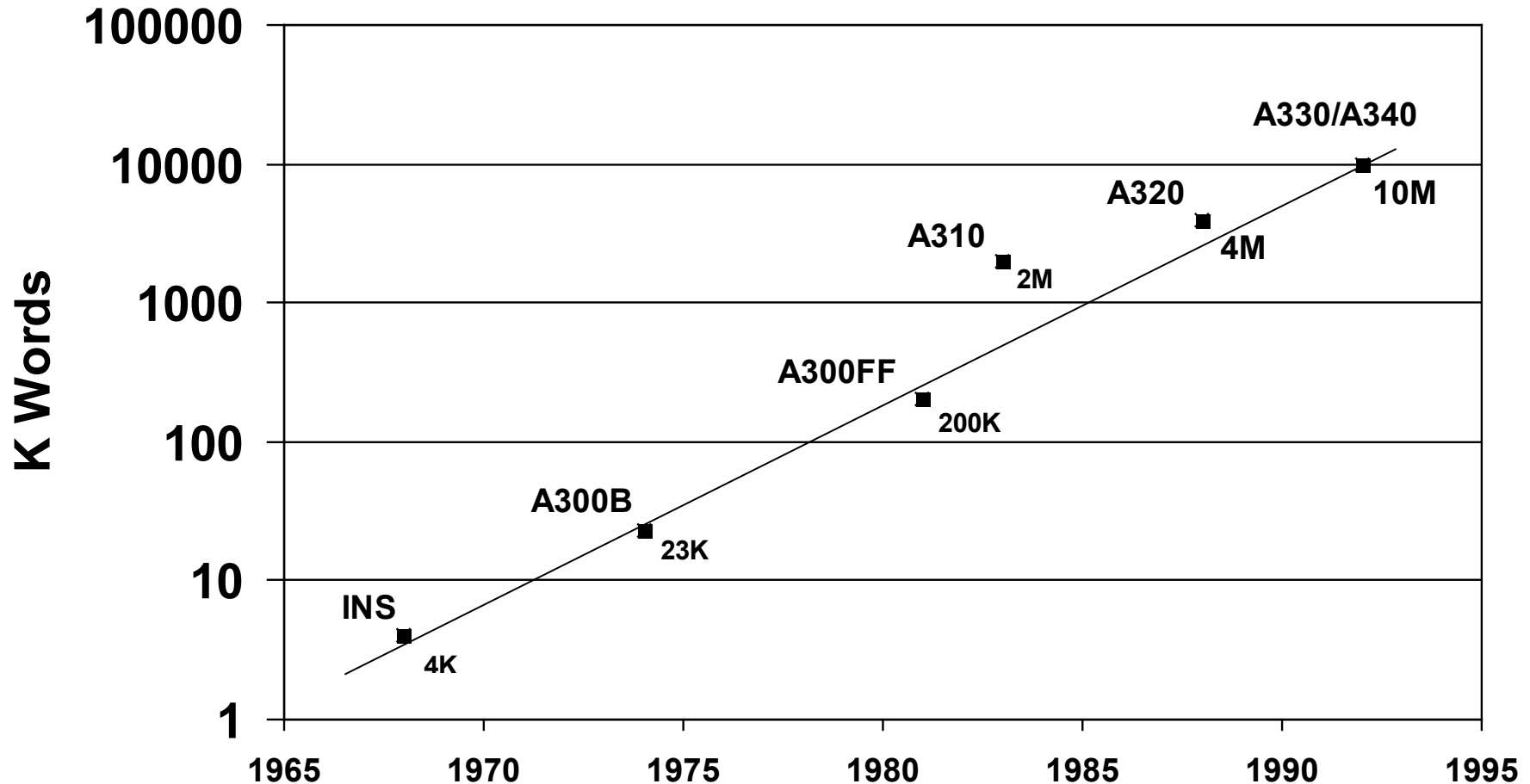
What Problem are We Solving?

Overview of Our Approach

Case Studies

Challenges and Future Directions

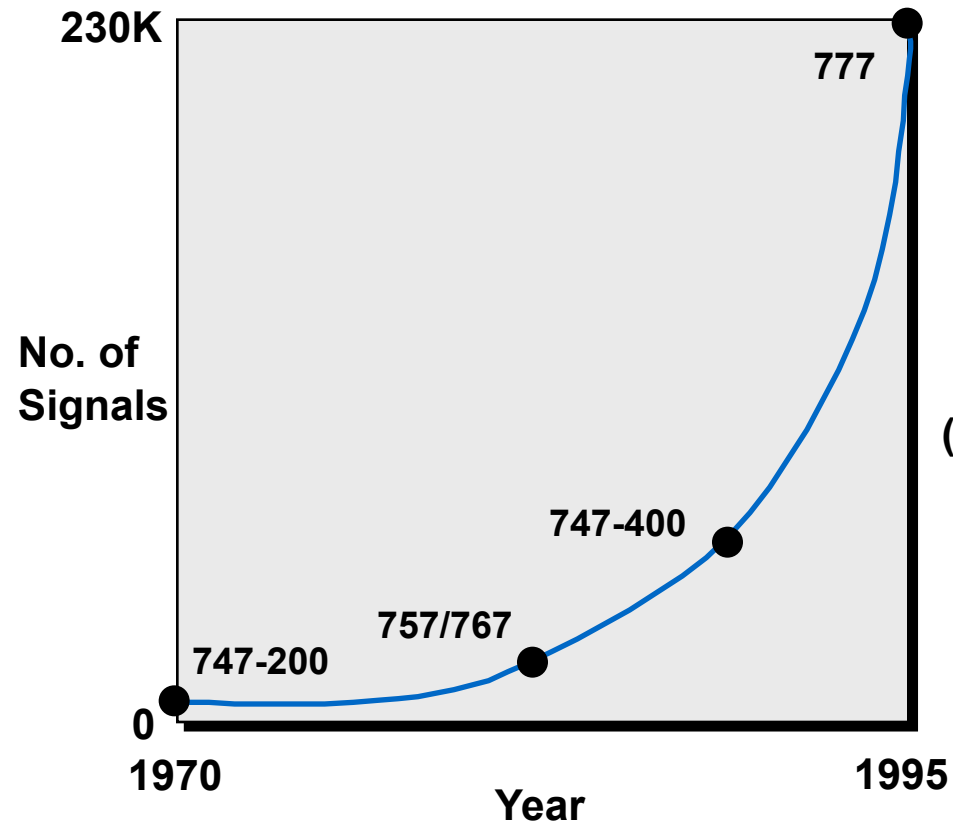
Airborne Software Doubles Every Two Years



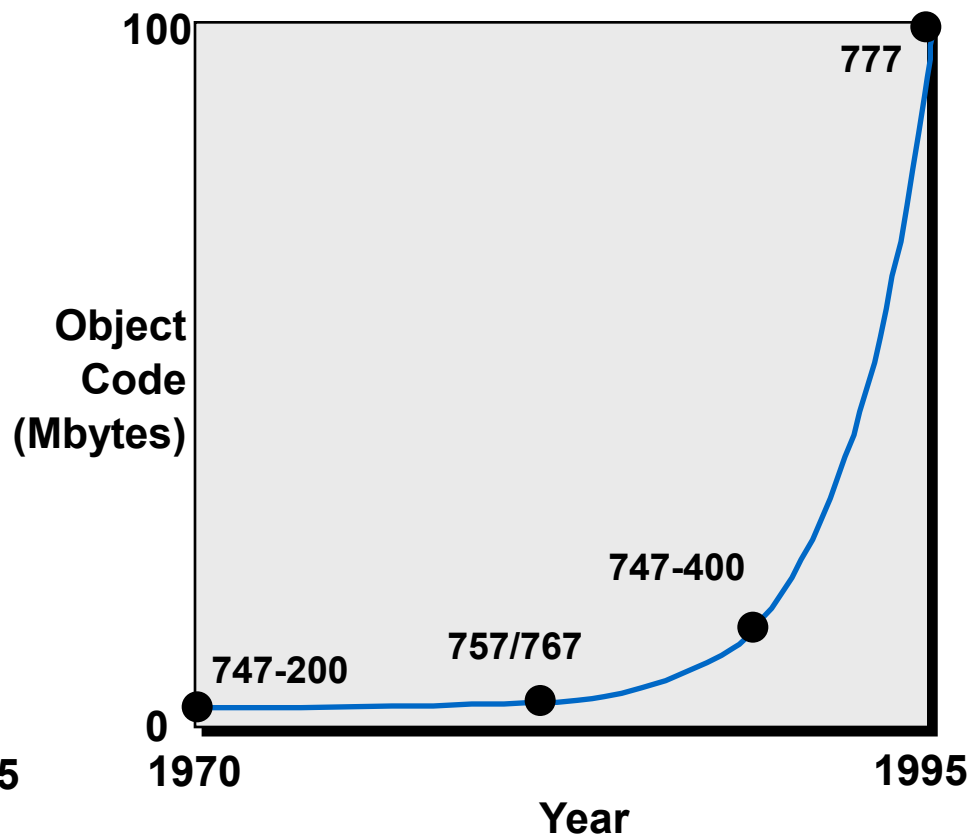
J.P. Potocki De Montalk, Computer Software in Civil Aircraft, Sixth Annual Conference on Computer Assurance (COMPASS '91), Gaithersburg, MD, June 24-27, 1991.

Similar Growth Has Been Seen by Boeing

Complexity



Size



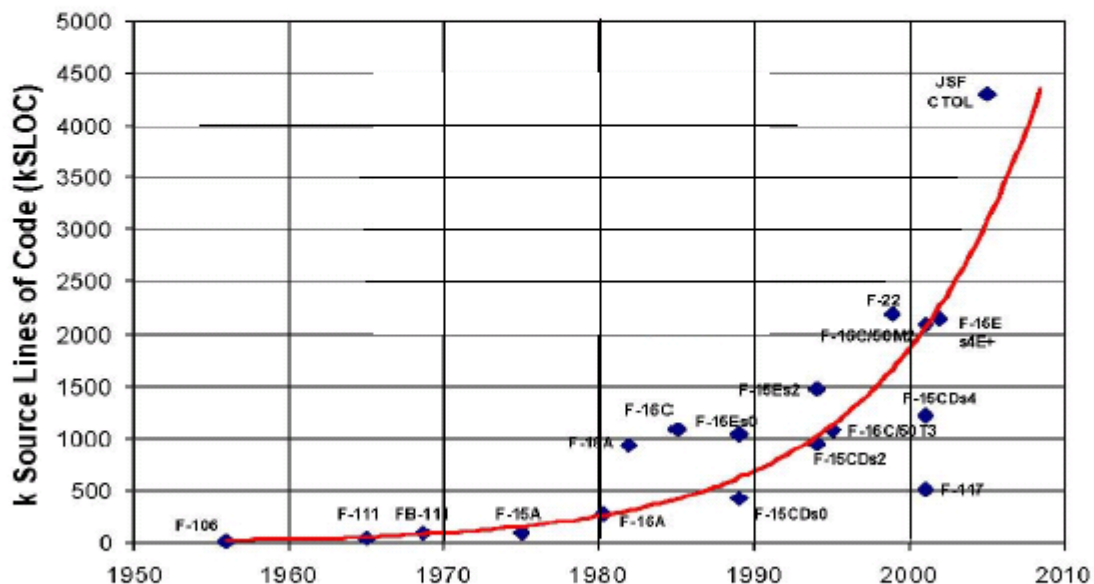


U.S. AIR FORCE

DoD software is growing in size and complexity



Total Onboard Computer Capacity (OFP)



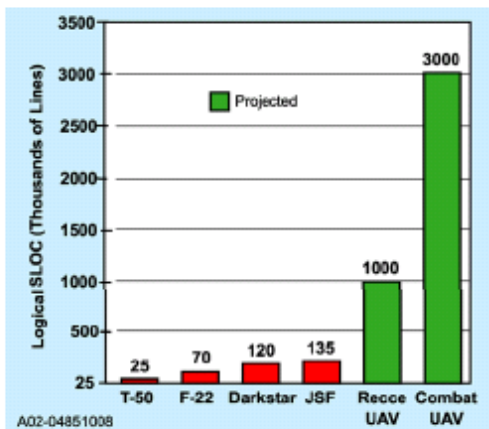
Source: "Avionics Acquisition, Production, and Sustainment: Lessons Learned -- The Hard Way", NDIA Systems Engineering Conference, Mr. D. Gary Van Oss, October 2002.

Robert Gold, OSD



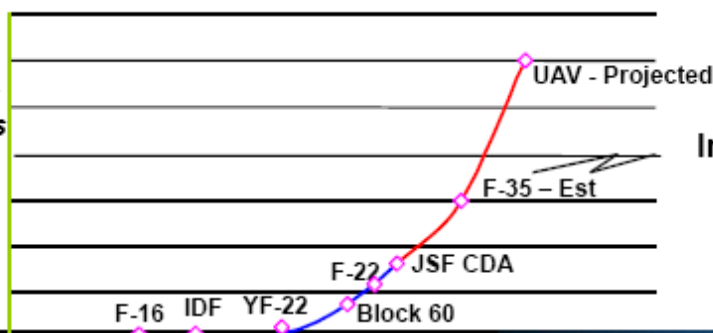
U.S. AIR FORCE

Emerging Software Size and Complexity



- Advanced system attributes (on-board *intelligence* and *adaptive control laws*) will be required to accommodate emerging functional requirements.
- This will increase the size and complexity of control systems beyond the capability of current V&V practices.

Inter-System
Communication
& Dependencies



Increasing system
integration
requirements and
Complexities

Projected Exponential Increase in SW Size and Complexity

Presentation Overview

Who Are We?

What Problem are We Solving?

 **Overview of Our Approach**

Case Studies

Challenges and Future Directions

Exploit the Convergence of Two Trends

- **Model-Based Development**
 - Domain specific graphical notations
 - MATLAB Simulink®, Esterel Technologies SCADE Suite™
 - Enable early simulation and debugging
 - Automated generation of code and tests
- **Model-Checking**
 - Prove properties about a model
 - Explore all possible inputs and states
 - Highly automated

Reduce Costs and Improve Quality by
Using Analysis to Find Errors During Early Design

Model-Based Development

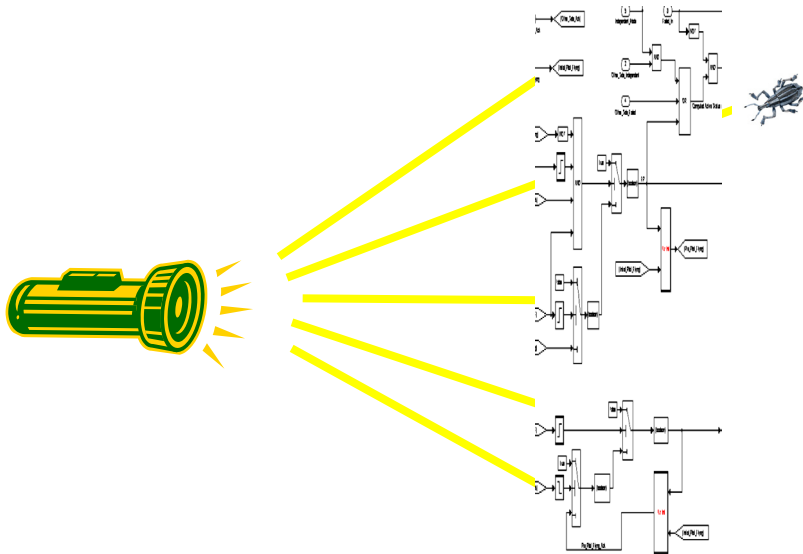
Company	Product	Tools	Specified & Autocoded	Benefits Claimed
Airbus	A340	SCADE With Code Generator	<ul style="list-style-type: none"> • 70% Fly-by-wire Controls • 70% Automatic Flight Controls • 50% Display Computer • 40% Warning & Maint Computer 	<ul style="list-style-type: none"> • 20X Reduction in Errors • Reduced Time to Market
Eurocopter	EC-155/135 Autopilot	SCADE With Code Generator	<ul style="list-style-type: none"> • 90 % of Autopilot 	<ul style="list-style-type: none"> • 50% Reduction in Cycle Time
GE & Lockheed Martin	FADEDC Engine Controls	ADI Beacon	<ul style="list-style-type: none"> • Not Stated 	<ul style="list-style-type: none"> • Reduction in Errors • 50% Reduction in Cycle Time • Decreased Cost
Schneider Electric	Nuclear Power Plant Safety Control	SCADE With Code Generator	<ul style="list-style-type: none"> • 200,000 SLOC Auto Generated from 1,200 Design Views 	<ul style="list-style-type: none"> • 8X Reduction in Errors while Complexity Increased 4x
US Spaceware	DCX Rocket	MATRIXx	<ul style="list-style-type: none"> • Not Stated 	<ul style="list-style-type: none"> • 50-75% Reduction in Cost • Reduced Schedule & Risk
PSA	Electrical Management System	SCADE With Code Generator	<ul style="list-style-type: none"> • 50% SLOC Auto Generated 	<ul style="list-style-type: none"> • 60% Reduction in Cycle Time • 5X Reduction in Errors
CSEE Transport	Subway Signaling System	SCADE With Code Generator	<ul style="list-style-type: none"> • 80,000 C SLOC Auto Generated 	<ul style="list-style-type: none"> • Improved Productivity from 20 to 300 SLOC/day
Honeywell Commercial Aviation Systems	Primus Epic Flight Control System	MATLAB Simulink	<ul style="list-style-type: none"> • 60% Automatic Flight Controls 	<ul style="list-style-type: none"> • 5X Increase in Productivity • No Coding Errors • Received FAA Certification

What Are Model Checkers?

- **Breakthrough Technology of the 1990's**
- **Widely Used in Hardware Verification (Intel, Motorola, IBM, ...)**
- **Several Different Types of Model Checkers**
 - **Explicit, Symbolic, Bounded, Infinite Bounded (SMT), ...**
- **Exhaustive Search of the Global State Space**
 - **Consider All Combinations of Inputs and States**
 - **Equivalent to Exhaustive Testing of the Model**
 - **Produces a Counter Example if a Property is Not True**
- **Easy to Use**
 - **"Push Button" Formal Methods**
 - **Very Little Human Effort Unless You're at the Tool's Limits**
- **Limitations**
 - **State Space Explosion (10^{100} – 10^{200} States)**

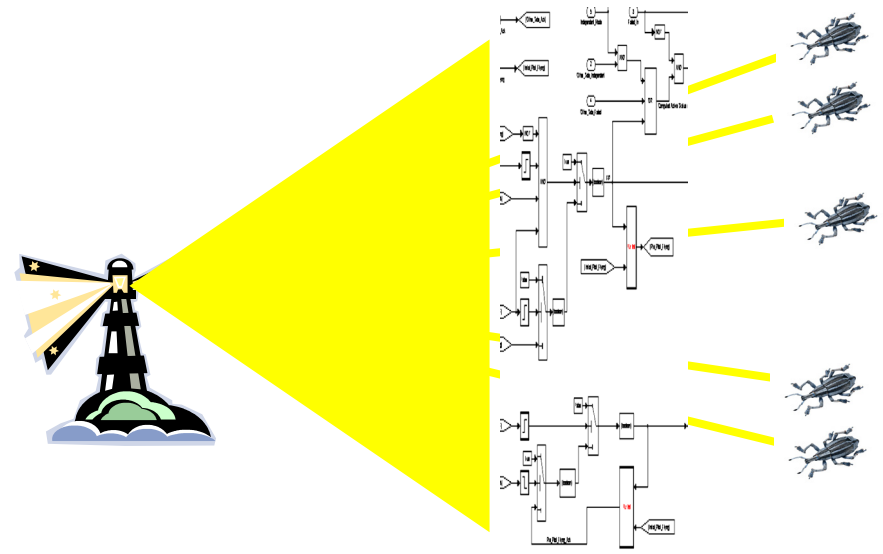
Advantage of Model Checking

Testing Checks Only the Values We Select



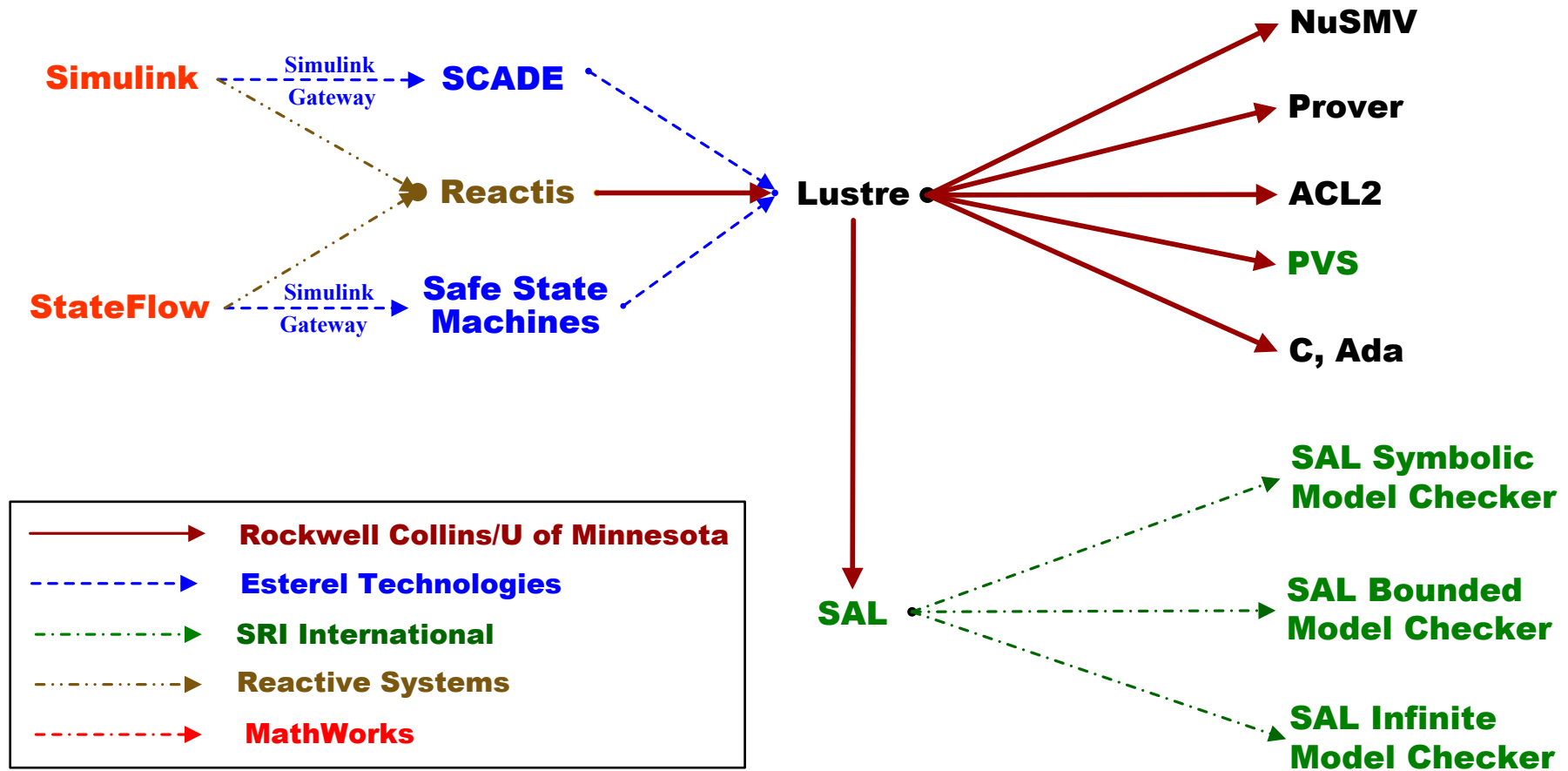
***Even Small Systems Have Trillions
(of Trillions) of Possible Tests!***

Model Checker Tries Every Possible Value!



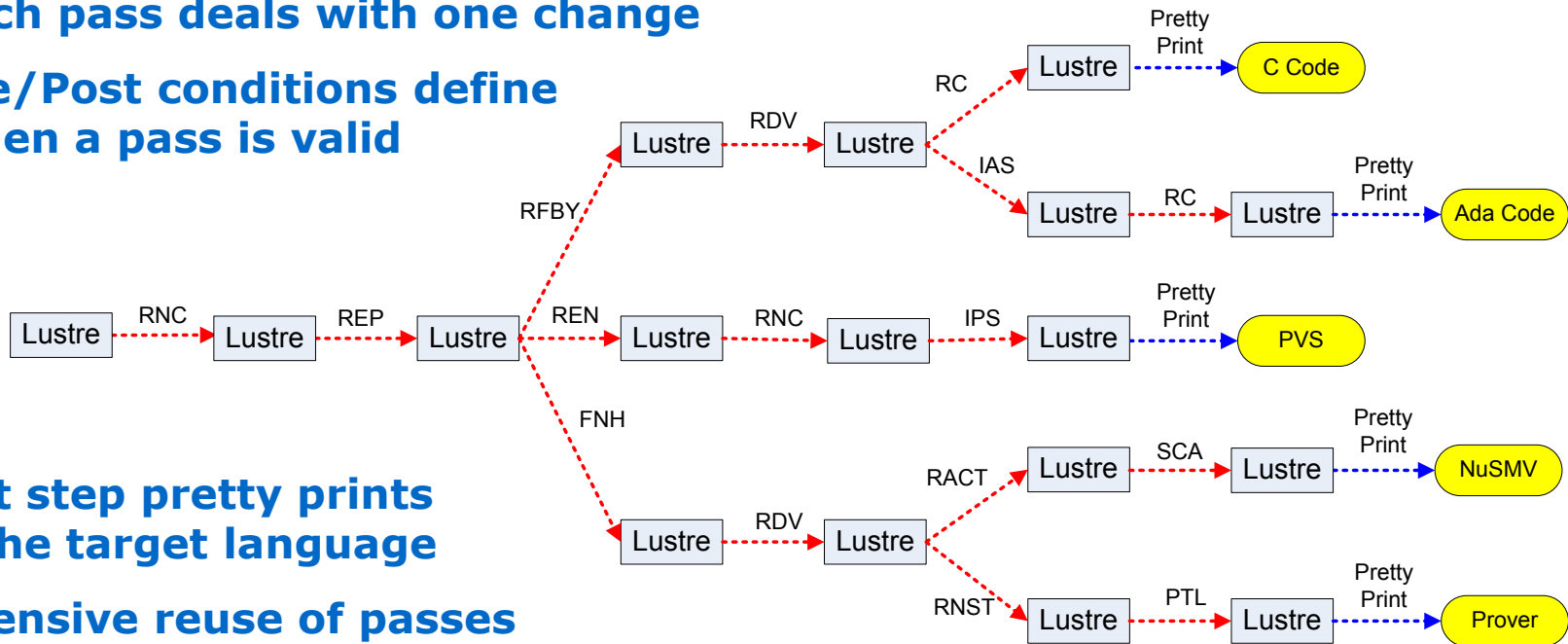
***Finds every exception to the
property being checked!***

Rockwell Collins Translation Framework



A Product Family of Translators

- Many small Lustre-to-Lustre translation passes
- Each pass refines closer to the target language
- Each pass deals with one change
- Pre/Post conditions define when a pass is valid



- Last step pretty prints to the target language
- Extensive reuse of passes
- New translators can be developed quickly (usually in less than a week)

Translators Optimize for Specific Analysis Tools

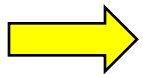
Model	CPU Time (For NuSMV to Compute Reachable States)		Improvement
	Before	After	
Mode1	> 2 hours	11 sec	> 650x
Mode2	> 6 hours	169 sec	> 125x
Mode3	> 2 hours	14 sec	> 500x
Mode4	8 minutes	< 1 sec	480x
Arch	34 sec	< 1 sec	34x
WBS	29+ hours	1 sec	105,240x

Presentation Overview

Who Are We?

What Problem are We Solving?

Overview of Our Approach



Case Studies

Challenges and Future Directions

ADGS-2100 Adaptive Display & Guidance System



Modeled in Simulink

Translated to NuSMV

4,295 Subsystems

16,117 Simulink Blocks

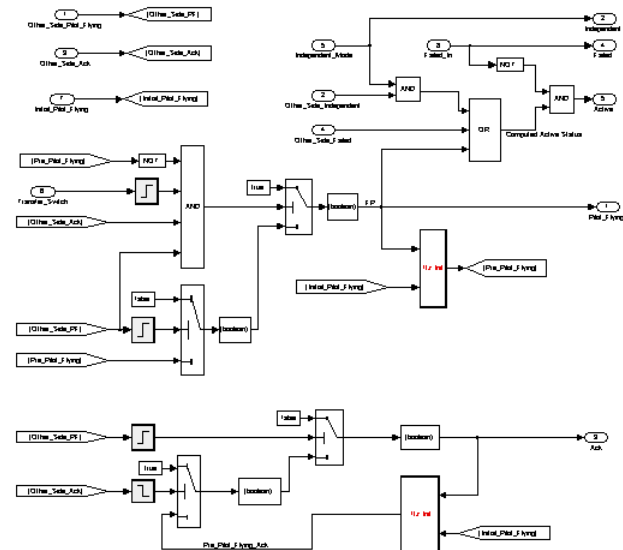
Over 10^{37} Reachable States

Example Requirement:

The Cursor Shall Never be
Positioned on an Inactive Display

Counterexample Found in 5 Seconds

Checked 563 Properties -
Found and Corrected 98 Errors
in Early Design Models

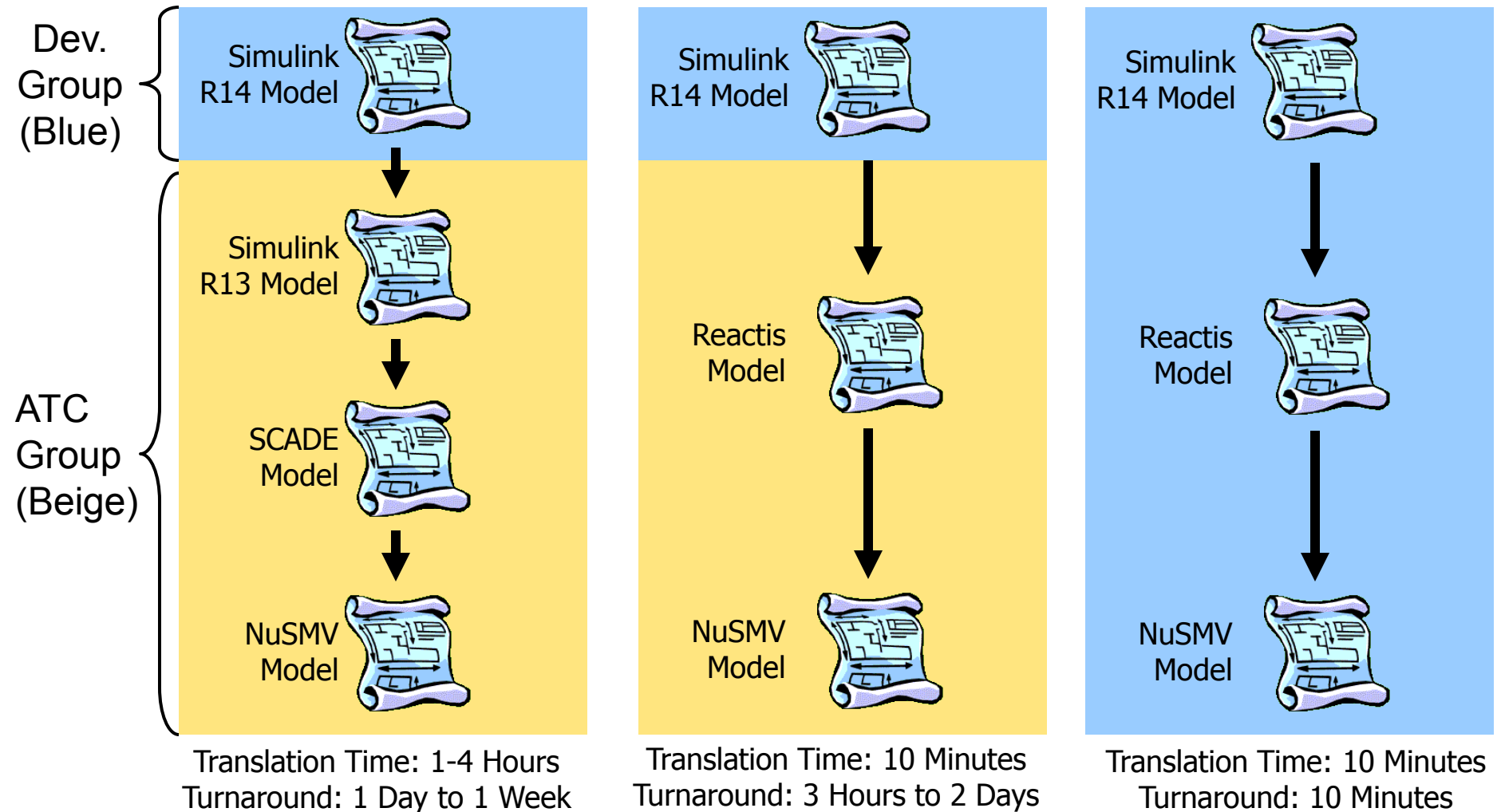


ADGS-2100 Technology Transfer

Iteration 1

Iteration 2

Iteration 3



CerTA FCS Phase I

- **Sponsored by the Air Force Research Labs**
 - **Air Vehicles (RB) Directorate - Wright Patterson**
- **Investigate Roles of Testing and Formal Verification**
 - **Can formal verification complement or replace some testing?**
- **Example Model – Lockheed Martin Adaptive UAV Flight Control System**
 - **Redundancy Management Logic in the Operational Flight Program (OFP)**
 - **Well suited for verification using the NuSMV model-checker**

Lockheed Martin Aero

- Based on Testing
- Enhanced During CerTA FCS
 - Graphical Viewer of Test Cases
 - Support for XML/XSLT Test Cases
 - Added C++ Oracle Framework
- Developed Tests from Requirements
- Executed Tests Cases on Test Rig

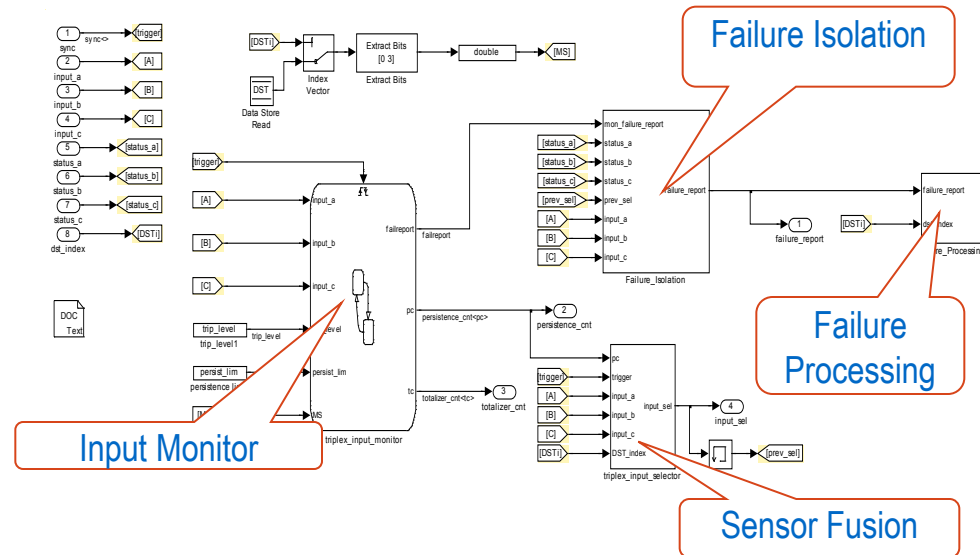
Rockwell Collins

- Based on Model-Checking
- Enhanced During CerTA FCS
 - Support for Simulink blocks
 - Support for Stateflow
 - Support for Prover model-checker
- Developed Properties from Requirements
- Proved Properties using Model-Checking

CerTA FCS Phase I - OFP Redundancy Management Logic

For Each of Ten Control Surfaces

- **Triplex Voter**
 - Input monitor, sensor fusion, and failure isolation
- **Failure Processing**
 - Logs failures into a data store
- **Reset Manager**
 - Reset logic for sensors and control surfaces (not shown)



	Subsystems / Blocks	Charts / Transitions	Truth Table Cells	Reachable State Space	Properties
Triplex voter	10 / 96	3 / 35	198	$6.0 * 10^{13}$	48
Failure processing	7 / 42	0 / 0	0	$2.1 * 10^4$	6
Reset manager	6 / 31	2 / 26	0	$1.32 * 10^{11}$	8
Total	23 / 169	5 / 61	198	N/A	62

CerTA FCS Phase I – Errors Found

	Model Checking	Testing
Triplex Voter	5	0
Failure Processing	3	0
Reset Manager	4	0
Total	12	0

- **Model-Checking Found 12 Errors that Testing Missed**
- **Spent More Time on Testing than Model-Checking**
 - **60% of total on testing vs. 40% on model-checking**

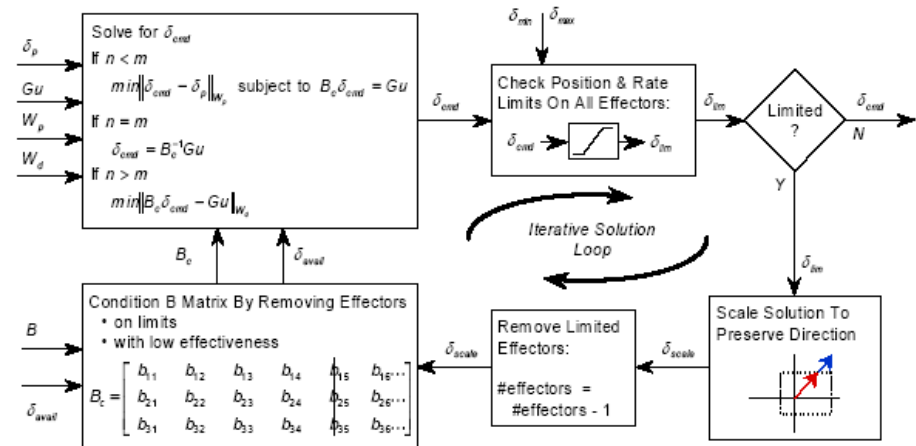
Model-checking was more cost effective than testing at finding design errors.

CerTA FCS Phase II

- **Sponsored by the Air Force Research Labs**
 - **Air Vehicles (RB) Directorate - Wright Patterson**
- **Can Model-Checking be Used on Infinite State Systems?**
 - **Large, numerically intensive, non-linear systems**

- **Example Model**

- **Lockheed Martin Adaptive UAV Flight Control System**
- **Effector Blender (EB)**
- **Generates actuator commands for aircraft control surfaces**
- **Matrix arithmetic of floating point numbers**

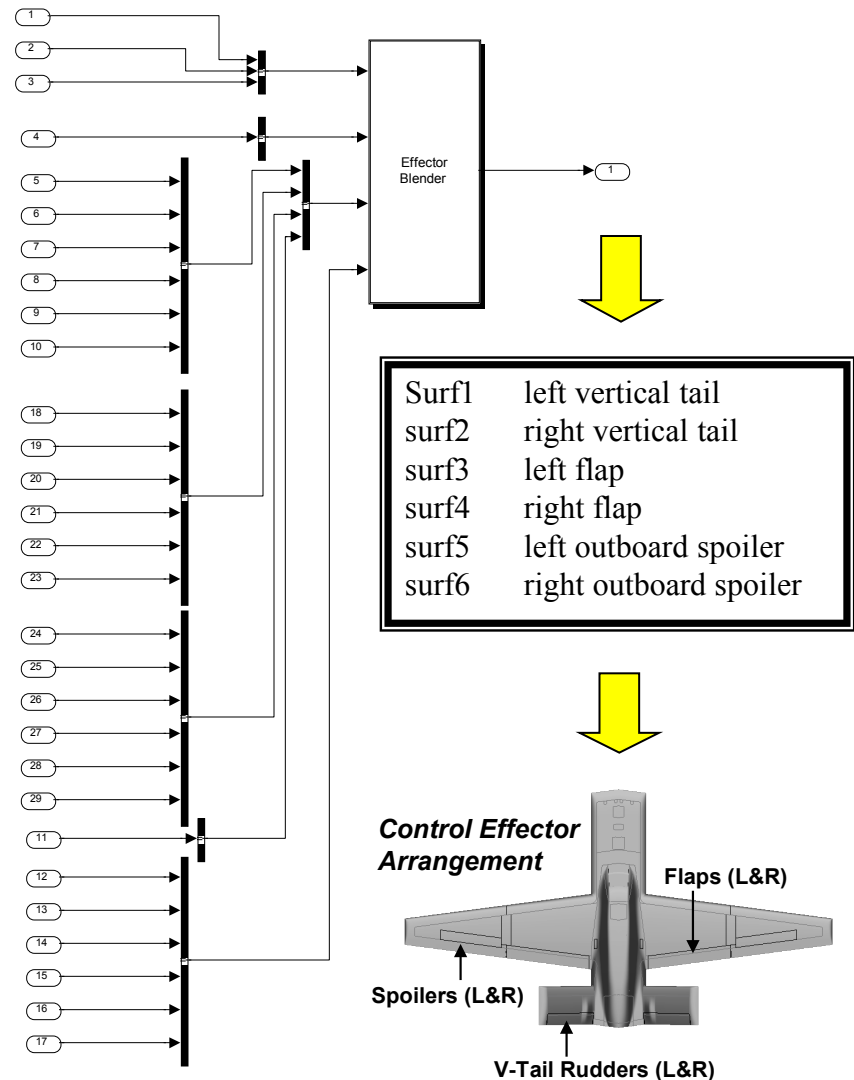


- **Challenges**

- **Identifying the right properties to verify**
- **Verification of floating point numbers**
- **Verification of Stateflow *flowcharts* with cyclic transition paths**
- **Compositional verification to scale to entire Effector Blender**

CerTA FCS Phase II – Effector Blender

- **Generates Actuator Commands**
 - Six control surfaces
 - Adapts its behavior as aircraft state changes
 - Iterative algorithm that repeatedly manipulates a 3 x 6 matrix of floating point numbers
- **Large Complex Model**
 - **Inputs**
 - 32 floating point inputs
 - 3 x 6 matrix of floating point values
 - **Outputs**
 - 1 x 6 vector of floating point values
 - **166 Simulink subsystems**
 - **2000+ basic Simulink blocks**
 - **Huge reachable state space**
- **Completely Functional**
 - **No internal state**



CerTA FCS Phase II – What to Verify?

- **No Explicit Requirements for the Effector Blender Model**
 - Requirements defined for Effector Blender + aircraft model
 - Addition of aircraft model pushes verification beyond current tools
- **Avoid Properties Verifiable by Other Means**
 - Control theory – stability, tracking performance, feedback design ...
 - Simulation – design validation
 - Implementation – code generation/compilation, scheduling, ...
- **Focus on the Consistency of the Effector Blender Model**
 - Relationships the model should always maintain
 - Partial requirements specification
- **Preservation of Control Surface Limits**
 - EB computes upper and lower limits for each control surface command
 - Function of aircraft design, aircraft state, and max extension per cycle
 - Commanded extension should always be between these limits

CerTA FCS Phase II – Verification of Floating Point Numbers

- **Floating Point Numbers**
 - Fixed number of bits with a movable decimal (radix) point
 - No decision procedures for floating point numbers available
- **Real Numbers**
 - Real numbers have unbounded size and precision
 - Would hide errors caused by limitations of floating point arithmetic
 - Control theory problems are inherently non-linear
 - Decision procedures for non-linear real numbers have exponential cost
- **Solution - Translate Floating Point Numbers into Fixed Point**
 - Extended translation framework to automate this translation
 - Convert floating point to fixed point (scaling provided by user)
 - Convert fixed point into integers (use bit shifting to preserve magnitude)
 - Shift from NuSMV (BDD-based) to Prover (SMT-solver) model checker
- **Advantages & Issues**
 - Use bit-level integer decision procedures for model checking
 - Results unsound due to loss of precision
 - Highly likely to find errors – very valuable tool for debugging

CerTA FCS Phase II - Results

- **Can Model-Checking be Used on Infinite State Systems?**

- Large, numerically intensive, non-linear systems

- **Effector Blender**

- **Inputs**

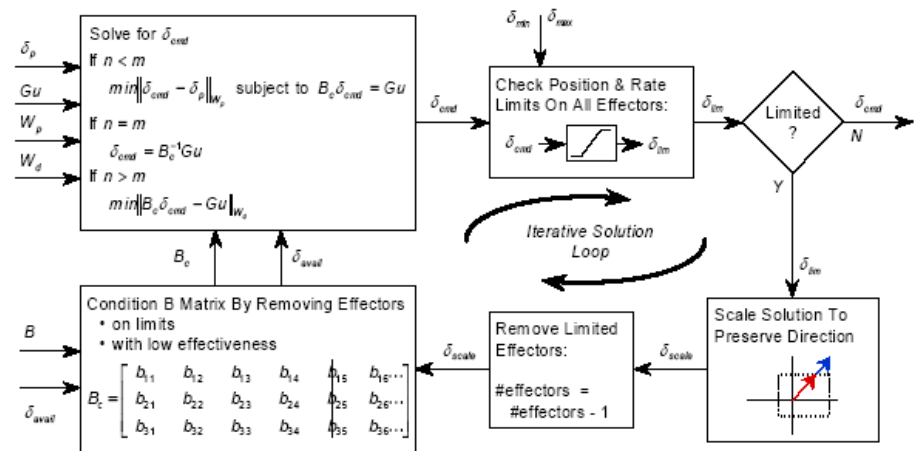
- 32 floating point inputs
- 3 x 6 matrix of floating point values

- **Outputs**

- 1 x 6 vector of floating point values

- **166 Simulink subsystems**

- **2000+ basic Simulink blocks**



- **Errors Found**

- Five previously unknown errors that would drive actuators past their limits
- Several implementation errors were being masked by defensive programming

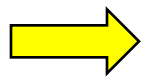
Presentation Overview

What Problem are We Solving?

Who Are We?

What are Formal Methods?

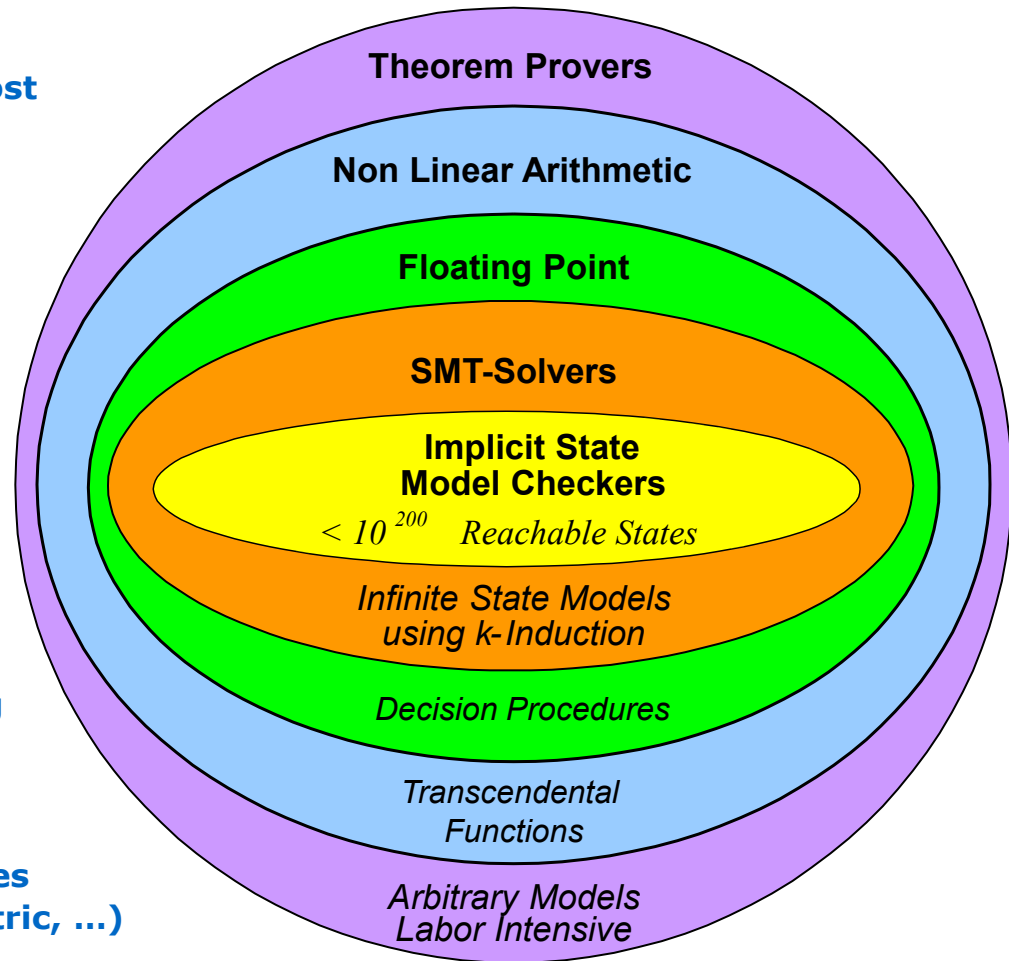
Examples of Using Formal Methods



Challenges and Future Directions

Extending the Verification Domain

- **Theorem Provers**
 - Deal with arbitrary models
 - Concerns are ease of use and labor cost
- **Large Finite Systems ($<10^{200}$ States)**
 - Implicit state (BDD) model checkers
 - Easy to use and very effective
- **Very Large or Infinite State Systems**
 - SMT-Solvers
 - Large integers and reals
 - Limited to linear arithmetic
 - Ease of use is a concern
- **Floating Point Arithmetic**
 - Most modeling languages use floating point (not real) numbers
- **Non-Linear Arithmetic**
 - Multiplication/division of real variables
 - Transcendental functions (trigonometric, ...)
 - Essential to navigation systems



Combining Theorem Proving and Model Checking For Compositional Verification

What Should the User Interface Be?

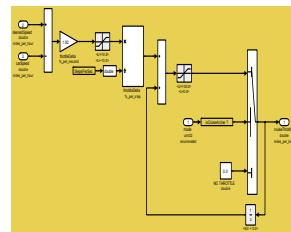
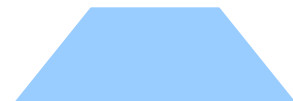
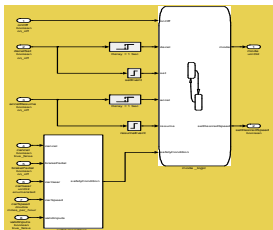
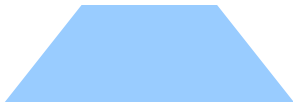
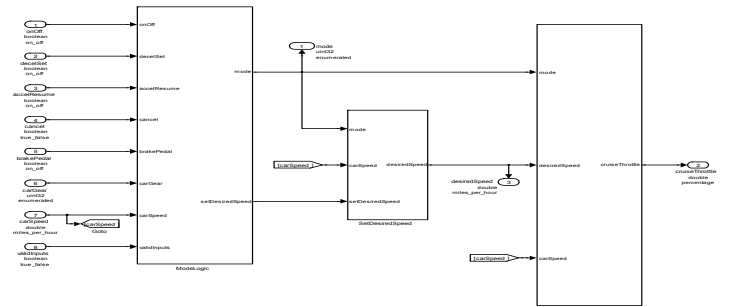
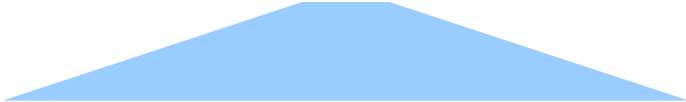
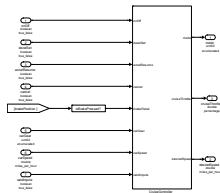
- Not emacs!
- Integrated with the model
- Simple theorem proving
- Powerful model checking

Composition of Subsystems

- Tends to be simple
- Well suited for theorem proving

Typical Model-Based Specification

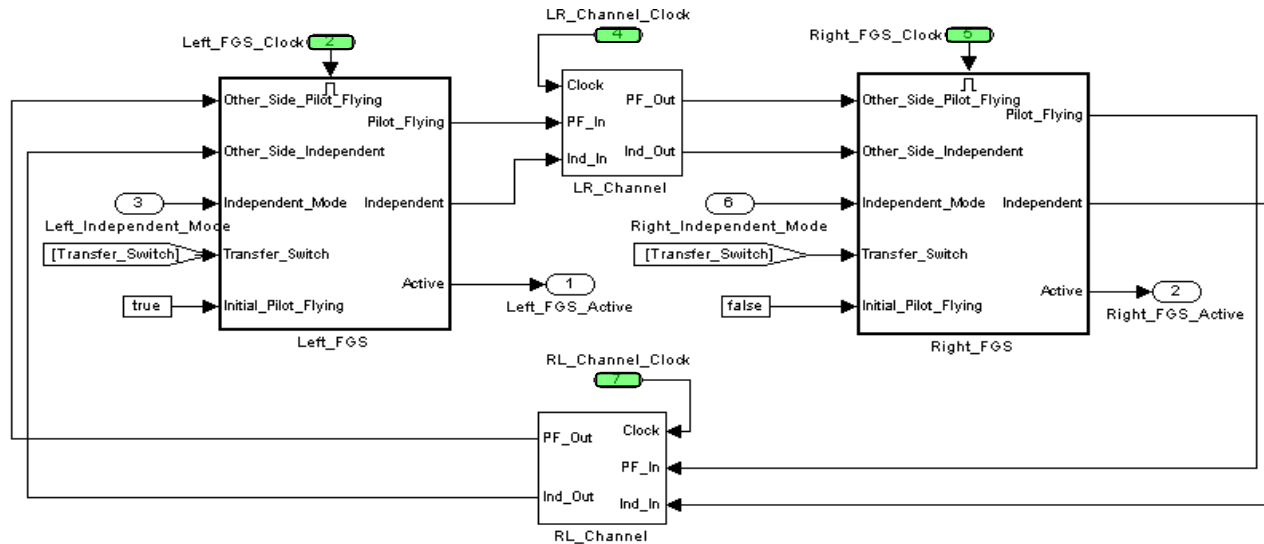
- Models are organized in a hierarchy several levels deep
- Most of the complexity is in the leaf models
- Leaf models can often be verified through model checking



Finding the Right Properties

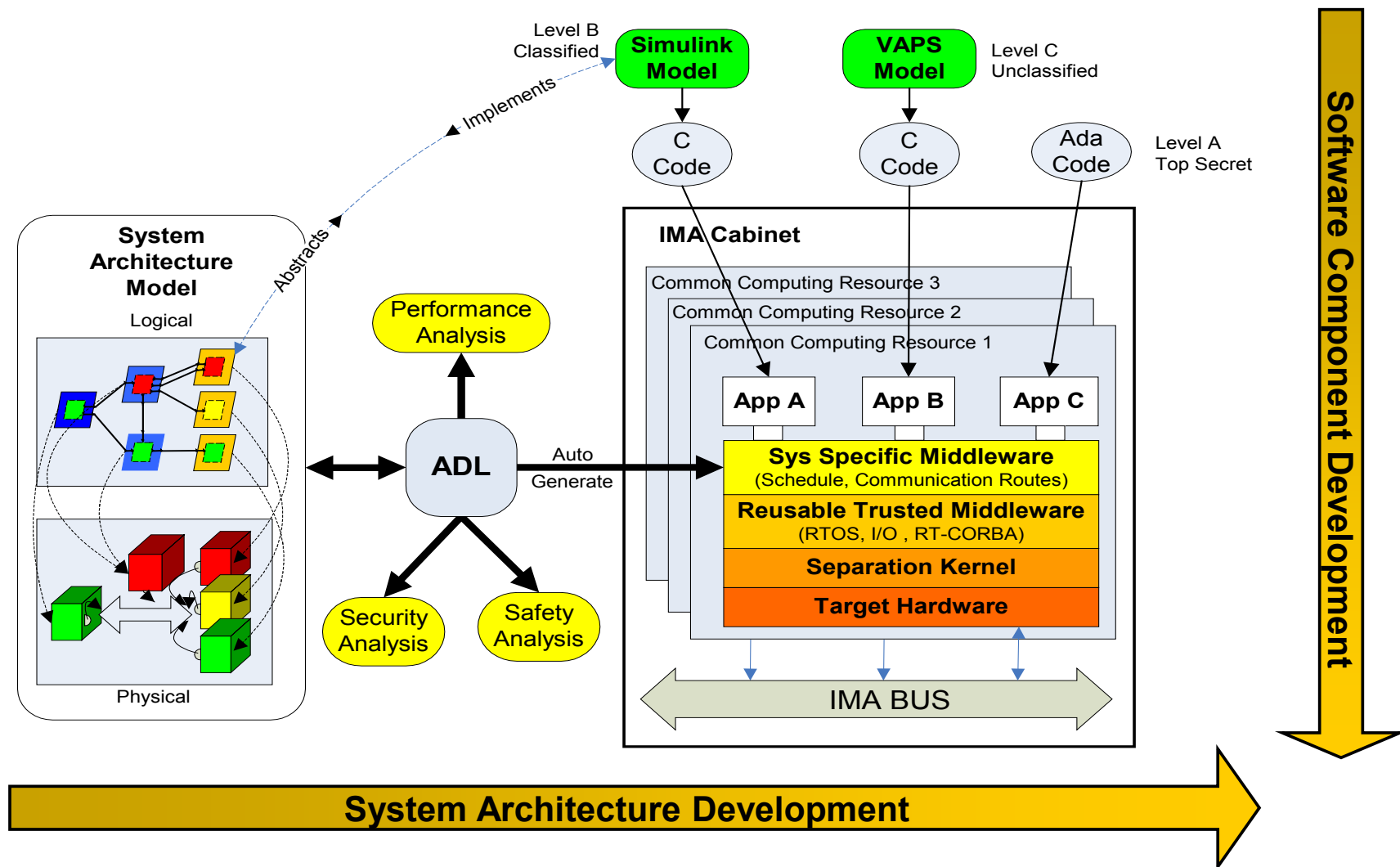
- **How Many Properties Do I Need?**
 - When am I done?
 - Are there coverage metrics like there are for testing?
 - How do I convince the certification authorities I'm done?
- **Are Some Properties Better than Others?**
 - Properties related to safety
 - Cross cutting properties find the most important errors
 - Simple, local properties find a surprising number of errors
- **Is There a Process or Heuristics for Defining Properties?**
 - Prove a property about each discontinuity in each output
- **What is the Relationship Between Proof and Testing**
 - Can proving replace some testing?
 - Can testing replace some proving?

Verifying Asynchronous Systems



- **Occur Frequently in System Designs**
 - **Implement fault tolerance or meet performance requirements**
- **No Global Clock - Each Node Has Its Own Clock**
- **Quasi-Synchronous System**
 - **Clocks have similar (but not identical) periods, drift, jitter**
- **Interleaving Leads to State Space Explosion**
 - **Makes model checking difficult**
- **Need to Exploit the Constraints Imposed by Quasi-Synchrony**

System Architectural Modeling & Analysis



Conclusions

- **Formal Methods Are Practical and Are Being Widely Used**
 - Model Based Development is the industrial face of formal methods
 - The engineers get to pick the modeling tools!
 - Semantics of some of the commercial tools could be improved
- **Formal Verification Tools Are Being Used in Industry**
 - Key is to verify the models the engineers are already building
 - Large portions of existing systems can be verified with model checkers
 - Need to make model checking accessible to the average engineer
- **Directions for the Future Work**
 - Making verification tools more powerful and easier to use
 - Integration of theorem proving and model checking
 - Finding the right properties
 - Verification of asynchronous systems
 - Modeling and analysis of system architectural models

For More Information

<http://shemesh.larc.nasa.gov/fm/fm-collins-intro.html>

- Whalen, M., Cofer, D., Miller, S., Krogh, B., Storm, W.: Integration of Formal Analysis into a Model-Based Software Development Process. In 12th International Workshop on Formal Methods for Industrial Critical Systems (FMICS2007), Berlin, Germany (2007).
- Whalen, M., Innis, J., Miller, S., Wagner, L.: ADGS-2100 Adaptive Display & Guidance System Window Manager Analysis, CR-2006-213952, NASA (2006).
- Mats P.E. Heimdahl, Michael W. Whalen, Ajitha Rajan, and Steven P. Miller, *Testing Strategies for Model-Based Development*, NASA Contractor Report NASA-2006-CR214307, April 2006. Available at <http://hdl.handle.net/2002/214307>.
- Miller, S., Tribble, A., Whalen, M., Heimdahl, M., Proving the Shalls, International Journal on Software Tools for Technology Transfer (STTT), Feb 2006.
- Michael W. Whalen, John D. Innis, Steven P. Miller, and Lucas G. Wagner, *ADGS-2100 Adaptive Display & Guidance System*, NASA Contractor Report NASA-2006-CR213952, Feb. 2006. Available at <http://hdl.handle.net/2002/16162>.
- Steven P. Miller, Mike W. Whalen, Dan O'Brien, Mats P.E. Heimdahl, and Anjali Joshi, A Methodology for the Design and Verification of Globally Asynchronous/Locally Synchronous Architectures, NASA Contractor Report NASA/CR-2005-213912, Sept. 2005. Available at <http://hdl.handle.net/2002/15934>.
- Miller, S., Anderson, E., Wagner, L., Whalen, M., Heimdahl, M.: Formal Verification of Flight Critical Software. In AIAA Guidance, Navigation and Control Conference and Exhibit, AIAA-2005-6431, American Institute of Aeronautics and Astronautics (2005).