Why We Model: Using MBD Effectively in Critical Domains

Mike Whalen
Program Director, UMSEC
University of Minnesota
Acknowledgements

- Rockwell Collins (Darren Cofer, Andrew Gacek, Steven Miller, Lucas Wagner)
- UPenn: (Insup Lee, Oleg Sokolsky)
- UMN (Mats P. E. Heimdahl)
- NASA Langley (Ricky Butler)
- Lockheed Martin (Walter Storm, Greg Tallant, Peter Stanfill)

Note: all incorrect or controversial opinions are mine only 😊
Outline of Presentation

Introduction

Why use Model-Based Development?
Requirements
Design
Implementation: Code Generation
Verification and Validation

Pitfalls
How we Develop Software

Concept Formation

Requirements Specification

Design

Implementation

Object Code

Analysis

Test
What is Model-Based Development?

Visualization

Properties

Analysis

Testing

Prototyping

Code/test generation

Specification Model

Test oracle

Code
Model-Based Development Tools

- Esterel Studio and SCADE Studio from Esterel Technologies
- Rhapsody from I-Logix
- Simulink and Stateflow from Mathworks Inc.
- Rose Real-Time from Rational
- I will focus on Statecharts and Dataflow notations.
How we **Will** Develop Software (in theory)
## Model-Based Development Examples

<table>
<thead>
<tr>
<th>Company</th>
<th>Product</th>
<th>Tools</th>
<th>Specified &amp; Autocoded</th>
<th>Benefits Claimed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airbus</td>
<td>A340</td>
<td>SCADE With Code Generator</td>
<td>• 70% Fly-by-wire Controls • 70% Automatic Flight Controls • 50% Display Computer • 40% Warning &amp; Maint Computer</td>
<td>• 20X Reduction in Errors • Reduced Time to Market</td>
</tr>
<tr>
<td>Eurocopter</td>
<td>EC-155/135 Autopilot</td>
<td>SCADE With Code Generator</td>
<td>• 90 % of Autopilot</td>
<td>• 50% Reduction in Cycle Time</td>
</tr>
<tr>
<td>GE &amp; Lockheed Martin</td>
<td>FADEDC Engine Controls</td>
<td>ADI Beacon</td>
<td>• Not Stated</td>
<td>• Reduction in Errors • 50% Reduction in Cycle Time • Decreased Cost</td>
</tr>
<tr>
<td>Schneider Electric</td>
<td>Nuclear Power Plant Safety Control</td>
<td>SCADE With Code Generator</td>
<td>• 200,000 SLOC Auto Generated from 1,200 Design Views</td>
<td>• 8X Reduction in Errors while Complexity Increased 4x</td>
</tr>
<tr>
<td>US Spaceware</td>
<td>DCX Rocket</td>
<td>MATRIXx</td>
<td>• Not Stated</td>
<td>• 50-75% Reduction in Cost • Reduced Schedule &amp; Risk</td>
</tr>
<tr>
<td>PSA</td>
<td>Electrical Management System</td>
<td>SCADE With Code Generator</td>
<td>• 50% SLOC Auto Generated</td>
<td>• 60% Reduction in Cycle Time • 5X Reduction in Errors</td>
</tr>
<tr>
<td>CSEE Transport</td>
<td>Subway Signaling System</td>
<td>SCADE With Code Generator</td>
<td>• 80,000 C SLOC Auto Generated</td>
<td>• Improved Productivity from 20 to 300 SLOC/day</td>
</tr>
<tr>
<td>Honeywell Commercial Aviation Systems</td>
<td>Primus Epic Flight Control System</td>
<td>MATLAB Simulink</td>
<td>• 60% Automatic Flight Controls</td>
<td>• 5X Increase in Productivity • No Coding Errors • Received FAA Certification</td>
</tr>
</tbody>
</table>

Slide courtesy of Steve Miller in "Proving the Shalls" © 2006 Rockwell Collins, Inc. All rights reserved.
Does Model-Based Development Scale?

Airbus A380

- Length: 239 ft 6 in
- Wingspan: 261 ft 10 in
- Maximum Takeoff Weight: 1,235,000 lbs
- Passengers: Up to 840
- Range: 9,383 miles

Systems Developed Using MBD
- Flight Control
- Auto Pilot
- Fight Warning
- Cockpit Display
- Fuel Management
- Landing Gear
- Braking
- Steering
- Anti-Icing
- Electrical Load Management

Slide courtesy of Steve Miller in “Proving the Shalls” © 2006 Rockwell Collins, Inc. All rights reserved.
…But it is not all roses

- Many MBD projects fail to meet their original goals of cost, productivity
  - These tend not to get as much publicity!
- Clear eyed understanding of why you model and what you expect is necessary
A Personal Anecdote

- Part of two large projects using Model-Based Development
  - Same company, similar quality developers
  - One great success
    - Significant cost reductions
    - Improvement in quality
    - Excellent customer satisfaction
  - One great failure
    - Large cost overruns
    - Models considered less useful than code
    - Group abandoned MBD
Outline of Presentation

Introduction

Why use Model-Based Development?
  Requirements
  Design
  Implementation: Code Generation
  Verification and Validation

Pitfalls
What are your models for?

- Possible to use MBD for many different purposes:
  - Requirements
  - Design
  - Simulation
  - Visualization
  - Testing
    - Test Generation
    - Test Oracle
  - Formal Verification
  - Code Generation
    - Complete implementation
    - Code skeleton
  - Prototyping
  - Communication with Customer

You must understand, **up front**, what you expect to do with models in order to successfully adopt MBD.

Major opportunity for improvement in V&V
MBD Models as Requirements

- Are MBD models requirements?

Notations in this talk are executable; good at describing how system works
- Lots of design detail
- Difficult to see “full system” behavior.
- Straightforward to generate code
Outline of Presentation

Introduction

Why use Model-Based Development?
  Requirements
  Design
  Implementation: Code Generation
  Verification and Validation

Pitfalls
The Most Important Issue for Successful Adoption of MBD

Do the Domain-Specific Notations provide a natural representation for your problem?

- Block diagrams are very natural for control problems
- Statecharts are very natural for description of system modes & mode transitions
- Both block diagrams and statecharts are very unnatural for representing complex data structures
- Neither notation naturally supports iteration or recursion
  - It can be “faked”, but not well
Just... No.

Stateflow model of Tetris game (included in the Stateflow Demo models from the Mathworks!).

Diagram is essentially a control-flow graph of a program that implements tetris.

*Much* harder to read and modify than an equivalent program.
Tools Matter

- Often notations are much more cumbersome to use than text
  - No diff / merge capabilities
  - Adding information requires many clicks
- Expressible != Easy
- Anecdote: Simulink vs. SCADE at Rockwell Collins in 2006
  - SCADE had formal pedigree, strong analysis
    - But tools kept crashing on our Windows boxes
  - Simulink had better tools and better salespeople
Outline of Presentation

Introduction

Why use Model-Based Development?
  Requirements
  Design
  Implementation: Code Generation
  Verification and Validation

Pitfalls
Analysis Pyramid

- Theorem Proving
- Perfect Verification
  Exhaustive Testing
  (Infinite Effort)
- Model Checking
  Temporal Properties
  Of Finite systems.
- MCDC Testing
- Data Flow Analysis
- Typical Testing
- Precise Analysis
  of Simple
  Syntactic
  Properties
- Simplistic
  Program
  Analysis
- Simplified
  Properties
- Optimistic
  Inaccuracy
- Pessimistic
  Inaccuracy

Pyramid Adopted from Dr. Michal Young

Typical Testing

Precise Analysis

of Simple

Syntactic

Properties

Data Flow

Analysis

MCDC

Testing

Perfect Verification

Exhaustive Testing

(Infinite Effort)

Model Checking

Temporal Properties

Of Finite systems.

Simplistic

Program Analysis

Pessimistic

Inaccuracy
What We Need

Model Checking Temporal Properties Of Finite Systems

Access to Many Tools and Techniques

Simplified Properties

Simplistic Program Analysis

Typical Testing

Data Flow Analysis

MCDC Testing

Precise Analysis of Simple Syntactic Properties

Perfect Verification

Exhaustive Testing (Infinite Effort)

Theorem Proving

Pessimistic Inaccuracy

Optimistic Inaccuracy
MBD Is a V&V-Enabling Technology

• Strong simulation and analysis capabilities built into most tools
  ◦ Demo: Stateflow Elevator
    • (Help: Stateflow/Demos/Large-Scale Modeling/Modeling an Elevator System)
• Even stronger simulation capabilities in external tools
  ◦ Demo: Reactis step simulation with Microwave
• Allows straightforward “Build a little, test a little” philosophy
  ◦ Consistent with incremental development philosophy
Model-Driven Test Generation (v1)

Source Code

```c
while(a<0) {
    a=a-1;
    b=b*a;
}
printf("%d", b);
```

Object Code

Test Case Generator

Compiler

Generated Tests

MBD Model

Coverage Metric

Possible to generate test suites that satisfy very rigorous structural coverage metrics

Model results must match source code for tests to pass
Model-Driven Test Generation (v2)

MBD Model

Test Case Generator

Code Generator + Compiler

Generated Tests

Object Code

Coverage Metric

Model should match source code exactly
Model-Driven Test Generation (v2)

Model should match source code exactly

Oracle

Where does Oracle come from? What is a good oracle?

Coverage Metric

Generated Tests

Test Case Generator

Code Generator + Compiler

Object Code

MBD Model

Model-Driven Test Generation (v2)
Use Requirements as Oracle

<table>
<thead>
<tr>
<th>Ref. #</th>
<th>English Requirements</th>
<th>SMV Proof</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>1 Mode Annunciations</strong></td>
<td></td>
</tr>
<tr>
<td>1.1</td>
<td><strong>1.1 Selection</strong></td>
<td></td>
</tr>
<tr>
<td>1.1.0-1</td>
<td>If this side is active and the mode annunciations are off, the mode annunciations shall be turned on when the onside FD is turned on.</td>
<td>SPEC AG((Mode_Annunciations_On &amp; IOnside_FD_On) -&gt; AX((ls_This_Side_Active = 1 &amp; Onside_FD_On) -&gt; Mode_Annunciations_On))</td>
</tr>
<tr>
<td>1.1.0-2</td>
<td>If this side is active and the mode annunciations are off, the mode annunciations shall be turned on when the offside FD is turned on.</td>
<td>SPEC AG((Mode_Annunciations_On &amp; Offside_FD_On = FALSE) -&gt; AX((ls_This_Side_Active = 1 &amp; Offside_FD_On = TRUE) -&gt; Mode_Annunciations_On))</td>
</tr>
<tr>
<td>1.1.0-3</td>
<td>If this side is active and the mode annunciations are off, the mode annunciations shall be turned on when the onside FD is turned on.</td>
<td>SPEC AG((Mode_Annunciations_On &amp; IOnside_FD_On) -&gt; AX((ls_This_Side_Active = 1 &amp; Onside_FD_On) -&gt; Mode_Annunciations_On))</td>
</tr>
<tr>
<td>1.2</td>
<td><strong>1.2 Deselection</strong></td>
<td></td>
</tr>
<tr>
<td>1.2.0-1</td>
<td>If this side is active and the mode annunciations are on, the mode annunciations shall be turned off if the onside FD is off, the offside FD is off, and the AP is disengaged.</td>
<td>SPEC AG(Mode_Annunciations_On -&gt; AX((ls_This_Side_Active = 1 &amp; IOnside_FD_On &amp; Offside_FD_On = FALSE &amp; IIs_AP_Engaged) -&gt; !Mode_Annunciations_On))</td>
</tr>
<tr>
<td>1.2.0-2</td>
<td>If this side is active and the mode annunciations are on, the mode annunciations shall not be turned off if the onside FD is on, or the offside FD is on, or the AP is engaged.</td>
<td>SPEC AG(Mode_Annunciations_On -&gt; AX((ls_This_Side_Active = 1 &amp; (Onside_FD_On</td>
</tr>
<tr>
<td>1.3</td>
<td><strong>1.3 Operation</strong></td>
<td></td>
</tr>
<tr>
<td>1.3.0-1</td>
<td>The mode annunciations shall not be on at system power up.</td>
<td>SPEC !(Mode_Annunciations_On)</td>
</tr>
<tr>
<td>1.3.0-2</td>
<td>If this side is active the mode annunciations shall be on if and only if the onside FD cues are displayed, or the offside FD cues are displayed, or the AP is engaged.</td>
<td>SPEC AG((ls_This_Side_Active = 1 -&gt; (Mode_Annunciations_On &lt;-&gt; (Onside_FD_On</td>
</tr>
</tbody>
</table>
Static Analysis and Model Checking

MDM Model

Oracle

Analysis Tool

Property True

Property False: Test Case
FCS 5000 Flight Control Mode Logic

Mode Controller A

Modeled in Simulink
Translated to NuSMV
6.8 x 10^{21} Reachable States

Mode Controller B

Example Requirement
Mode A1 => Mode B1
Counterexample Found in
Less than Two Minutes
Found 27 Errors

Slide © Rockwell Collins, 2008
Example Requirement:
Drive the Maximum Number of Display Units
Given the Available Graphics Processors

Counterexample Found in 5 Seconds

Checked 573 Properties -
Found and Corrected 98 Errors
in Early Design Models

Modeled in Simulink
Translated to NuSMV
4,295 Subsystems
16,117 Simulink Blocks
Over $10^{37}$ Reachable States
CerTA FCS Phase I

- Sponsored by AFRL
  - Wright Patterson VA Directorate
- Compare FM & Testing
  - Testing team & FM team
- Lockheed Martin UAV
  - Adaptive Flight Control System
  - Redundancy Management Logic
  - Modeled in Simulink
  - Translated to NuSMV model checker

Phase I Results

<table>
<thead>
<tr>
<th>Effort (% total)</th>
<th>Errors Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testing</td>
<td>60%</td>
</tr>
<tr>
<td>Model-Checking</td>
<td>40%</td>
</tr>
</tbody>
</table>
MBD Formal Analysis Efforts

Examples of Using Formal Methods

Turnstile High Integrity Guard
- High-assurance cross domain platform that provides secure communication between different security classification domains ranging from top secret to unclassified.
- Core guard application is based on the NSA certified AAMP7G.
- I/O processing is relegated to Offload Engines (OE) that do not have to be as highly trusted.
- System integrator can add function to the OE without compromising the guard function.
- Certification based on ACL2 theorem prover

Formal Analysis of a Triplex Sensor Voter in an Industrial Context

Michael Dierkes
Rockwell Collins France

FMICS 2011 workshop
August 30, 2011
Trento
Outline of Presentation

Introduction

Why use Model-Based Development?
  Requirements
  Design
  Implementation: Code Generation
  Verification and Validation

Pitfalls
Problem 1: Using Models Where They Don’t Fit

If MBD notation doesn’t provide a better representation of your problem than code, you’re wasting your time.
MBD notations can be awful programming languages…
Remedies

- Perform honest assessment of where MBD notations can be used
  - They do not do everything
  - Recursive data structures are especially difficult to model.
  - Use models where they are a good representation.

- Create a partitioning strategy between models and code for applications that contain both complex mode logic and complex data.
Problem 2
Believing Testing Can be Eliminated

Testing will always be a crucial (and costly) component
Testing Does not go Away

Concept Formation

Requirements

Properties

Specification/Model

Implementation

System

Extensive Testing (MC/DC)
It Simply Moves

Concept Formation

Requirements

Specification/Model

Properties

Implementation

System

Extensive Testing (MC/DC)
Do it the Right Way

Concept Formation → Requirements → Specification/Model → Implementation → Unit Test → System Test

Analysis → System

Integration Test
Problem 3
Believing the Model is Everything

The model is never enough
Modeling Frenzy

Concept Formation

Specification/Model

How do we know the model is “right”?

Implementation

Test

Modeling is so much fun
Remedies

- **Recognize the Role of Software Requirements**
  - The model is not everything

- Development Methods for Model-Based Development Badly Needed
  - Model-Based Software Development Process

- Develop Tools and Techniques for Model, Properties, and Requirements Management

- Develop Inspection Checklists and Style Guidelines for Models
Problem 4

Trusting Verification

To really mess things up, you need formal verification
The Mode Annunciations shall be turned on when the Flight Director is turned on

\textbf{AG}(\textit{Onside\_FD\_On} \rightarrow \textit{Mode\_Annunciations\_On})

If this side is active, the Mode Annunciations shall be turned on when the Flight Director is turned on

\textbf{AG}( (\textit{Is\_This\_Side\_Active} \& \textit{Onside\_FD\_On}) \rightarrow \textit{Mode\_Annunciations\_On})

If this side is active and the Mode Annunciations are off, the Mode Annunciations shall be turned on when the Flight Director is turned on

\textbf{AG}( ! \textit{Mode\_Annunciations\_On} \rightarrow \textbf{AX} ((\textit{Is\_This\_Side\_Active} \& \textit{Onside\_FD\_On}) \rightarrow \textit{Mode\_Annunciations\_On}))
Remedies

- Develop techniques to determine adequacy of model and property set
  - How do we know they are any “good”
- Techniques for management of invariants
  - How do we validate the assumptions we make
- Methodology and guidance badly needed
  - Tools with training wheels
  - “Verification for Dummies”

All we need is one high-profile verified system to fail spectacularly to set us back a decade or more
Conclusions

- MBD can significantly improve developer productivity, cost, schedule, and quality
- …or it can make your life miserable
- The important thing is to **know why you’re doing it!**
  - Know the limitations of what can be modeled using the DSNs
  - Know which capabilities you hope to use
    - Design and quality of models depends on this
- V & V receives the largest benefit of the MBD approach
  - Mature tools for test-case generation
  - Starting to see model checking built into commercial tools: SCADE Verifier, Simulink Design Verifier
- There are many other things to discuss! Versioning, diff, semantics, tool costs, training, structuring, vendor “lock in”
Questions?
References


## Medical Cyber-Physical Systems

Improving patient treatment by coordinated systems of medical devices

### Research directions:
- Medical device interoperability
- High-confidence development
  - Model-driven design
  - V&V, regulatory approval

**Supported by NSF CNS-1035715**

http://rtg.cis.upenn.edu/MDCPS/

### Participants
- University of Pennsylvania
- U. Penn Hospital System
- University of Minnesota
- CIMIT/MGH

---

### Coordination framework for medical devices
- Build high-confidence middleware
  - Rely on formal methods and static analysis
- Design a language for executable clinical scenarios
  - Specify information flows
  - Identify timing constraints
  - Ensure non-interference

### Model driven development and assurance cases
- High-assurance development:
  - Modeling, code synthesis
  - Model-level verification, code-level validation
- Assurance case construction reflects development process structure
- Applied to pacemaker, PCA pump

---

### Networked Blood Glucose Control System

Safety-critical, closed-loop MCPS

**Research issues:**
- Identifying new risks and hazards
- Mitigation strategies
- Validation
- Control design
- Pursue model-driven approach

---

### Smart alarm systems
- Reduction of irrelevant alarms for CABG patients
  - Based on aggregation of multiple vital signs and fuzzy logic
- On-going research:
  - Prediction of vasospasm in neuro-ICU patients

---

5/19/2013

Why We Model - Mike Whalen