Raytheon

Customer Success Is Our Mission

Using Model-driven Engineering Techniques for Integrated Flight Simulation Development

Douglas Fiehler
Brett Collins
Jesse Carlaftes
Raytheon Missile Systems

October 29, 2009

Outline

Raytheon
Missile Systems

- Introduction of Model-driven Engineering (MDE)
- History of MDE at Raytheon Missile Systems
- Intentions of Using MDE for Integrated Flight Simulation (IFS) Development
- MDE Tool History Example
- Model Lifecycle Comparison
- MDE Process Flow
- Time Savings Comparison
- Performance in Integrated Flight Simulations
- Common Pitfalls
- Conclusions

Introduction of Model-driven Engineering

Raytheon Missile Systems

- Model-driven Engineering
  - A.K.A. Model-driven Development (MDD)
  - Software development methodology that focuses on creating models rather than algorithms
  - Domain experts maintain more control of the software end product
  - Promotes compatibility and communication between individuals/teams
- One Tool's Role in MDE
  - Simulink® is a Popular tool for domain experts' development of system models
  - Real-Time Workshop® Embedded Coder provides MDE interface to Integrated Flight Simulations (IFSs) through automatic generation of C/C++ code
  - IFS engineer owns process of creating code

Real-Time Workshop® provides an MDE interface to the IFS

History of MDE at Raytheon Missile Systems

Raytheon Missile Systems

- Initial work
  - Automatic code generation process created to support rapid algorithm development
    - Identified limitations and pitfalls
    - Standardized deployment for incorporation in object oriented simulations
  - Original Processes developed using release Matlab® R11
- Ongoing efforts
  - Process has been implemented on many programs
    - Hardware models
    - Control algorithms
    - Medium and high fidelity
  - Presently using Matlab® Release 2009a
    - Processes updated for current releases

MDE Processes are in place and are being used at Raytheon Missile Systems.

Intentions of Using MDE for Integrated Flight Simulation Development

Raytheon Missile Systems

- MDE is a powerful process for designing models, both hardware and software, for simulations
  - Because of requirements imposed on IFSs, impractical to develop entire simulation with MDE
- Early development of IFSs requires frequent changes to models
  - Automatic code generation from MDE methods saves time, not only in initial integration of the model into the IFS, but subsequent changes can be made simpler and quicker.
- While much initial model design work done with Simulink®, other MDE tools are used to develop flight software

MDE, when used appropriately, is a powerful tool for IFS development

MDE Tool History Example

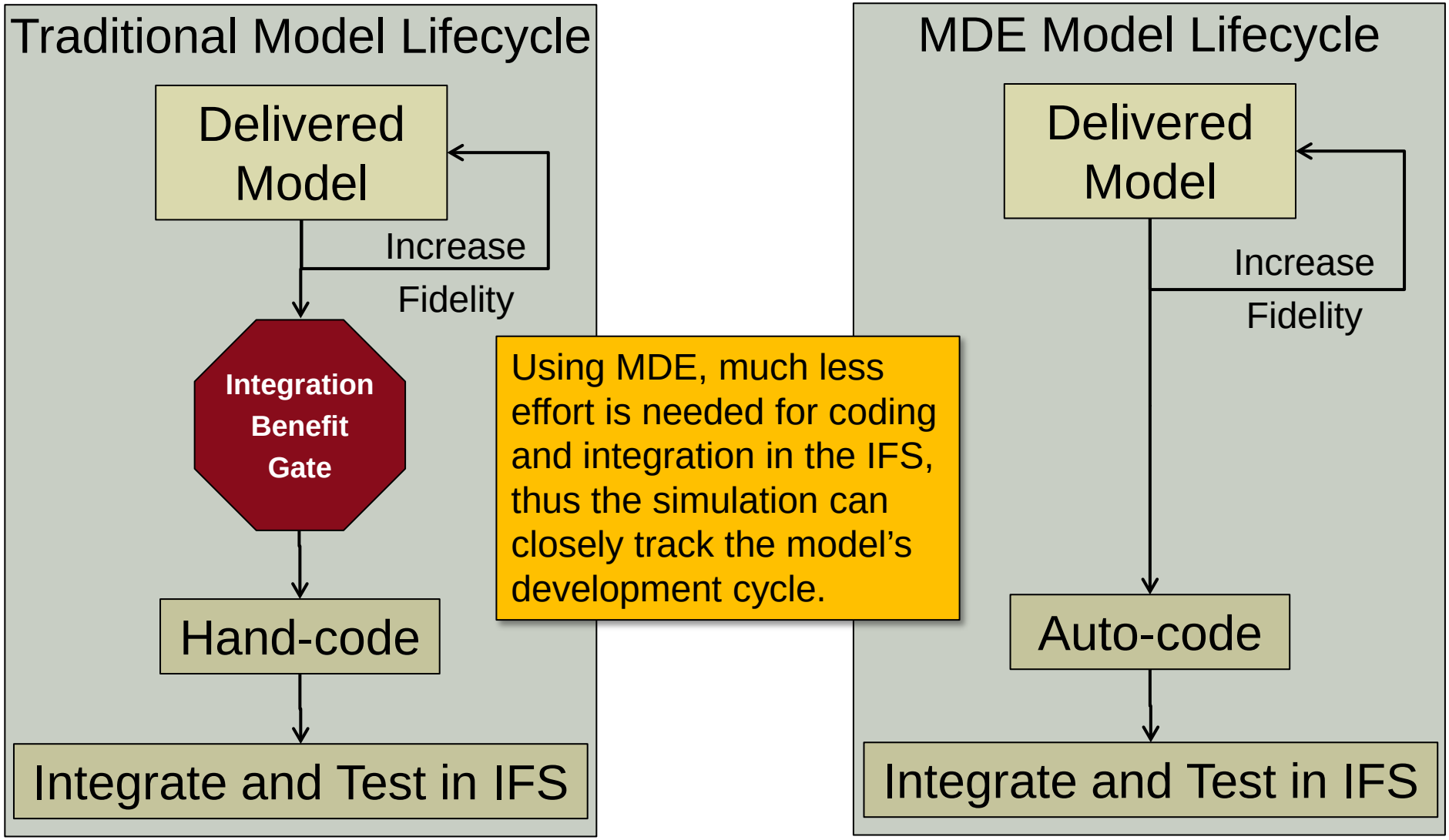
Raytheon
Missile Systems

- MDE Tools Evolve Over Time, and so must MDE processes
  - Matlab® R2008a and Previous
    - Would generate only C code
    - C++ option only changed the file extensions from ".c" to ".cpp"
    - Early versions (R11) could only support discrete models
  - Releases Since Matlab® R2008b
    - Includes option to generate "Encapsulated C++" code
    - True C++ class that can be instantiated in the IFS (multiple times if needed)
    - Includes Initialize, Step, and Finalize member functions
    - Additional member functions for setting or getting static input variables

Continuous MDE tool improvements require process improvements

Model Lifecycle Comparison

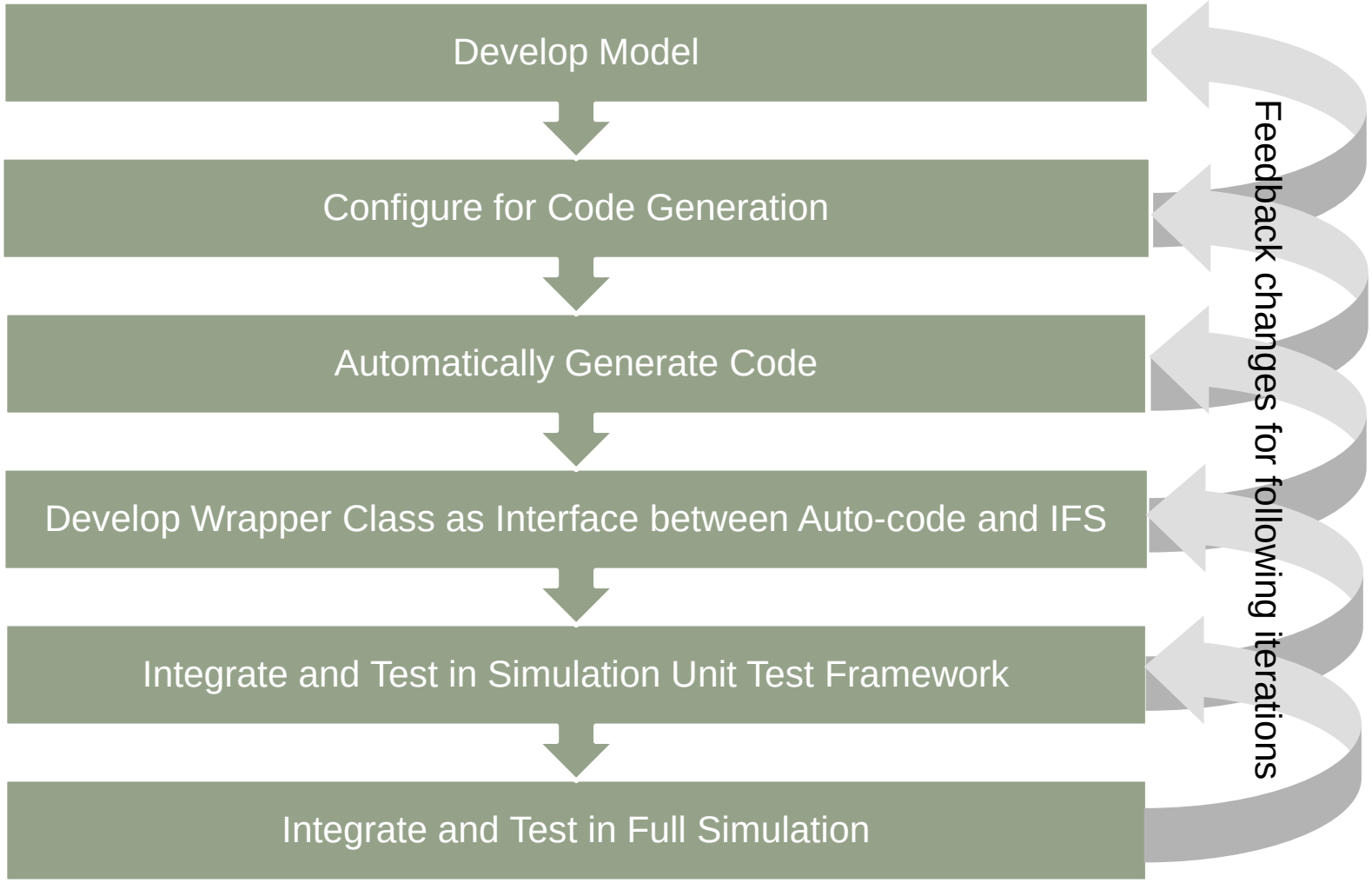
Raytheon
Missile Systems



Autocoding can reduce cycle time for integrating updated models

MDE Process Flow

Raytheon
Missile Systems



Straightforward process using MDE models to develop Functional Simulations

Time Savings Comparison

Raytheon
Missile Systems

- Hardware model coded
  - Control Actuation System model
  - Representative model for most hardware models integrated in IFS
- Used three methods to obtain time comparisons
  - Hand-coded from Simulink® block diagram
  - Auto-coded using original process using Matlab® R11
    - Can only use discrete blocks and integration when auto-coding
  - Auto-coded using updated process using Matlab® R2008b
    - Continuous blocks and integration supported
- Note that process times are for a first pass through the auto-coding process
  - Subsequent integrations of the same model should show even further process time reductions

Time Savings Comparison

Raytheon Missile Systems

Task	Hand-coding (hr)	Auto-code without Continuous Block Support (hr)	Auto-code with Continuous Block Support (hr)
Create usable source code from using MDE			
Insert and connect generic I/O port content		2	2
Replace Integrators with ports		2	
Continuous block identification and replacement		8	
Auto-code option selection and code generation		<1	<1
Preparation of generated code		4	4
Handcoding model - Simulation	60		
Handcoding model – Algorithm Design Tools	60		
Subtotal	**120**	**17**	**7**
Common efforts to integrate code into IFS			
Modifying IFS wrapper object, input files, etc	4	4	4
Performing unit Tests for verification	4	4	4
Performing Simulation Tests for verification	10	10	10
Subtotal	**18**	**18**	**18**
Total Conversion Time	**138**	**35**	**25**
% of Hand-coding	**100%**	**25.4%**	**18.1%**

Significant time savings when auto-coding models

Performance in Integrated Flight Simulations

Raytheon Missile Systems

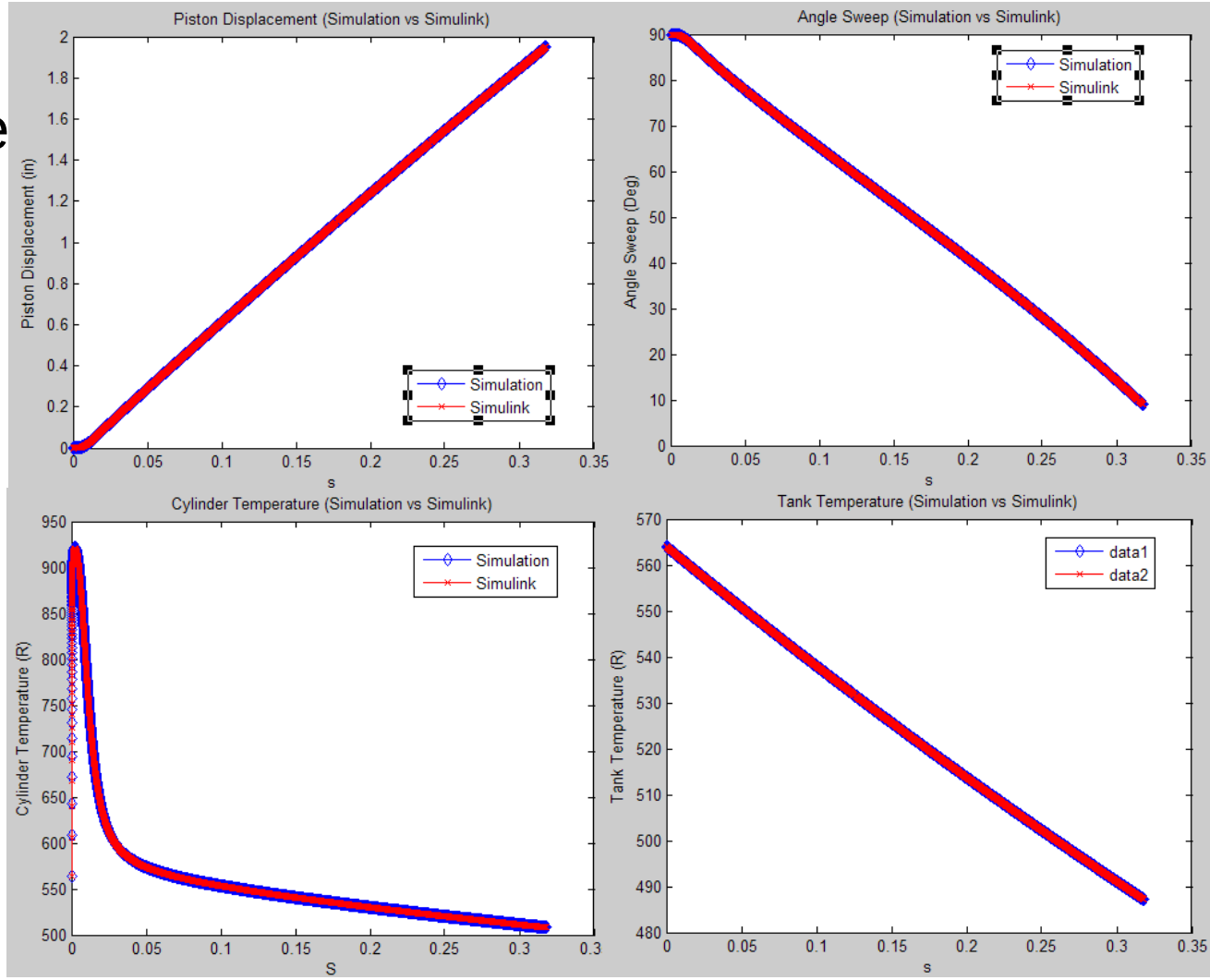
- Currently using MDE processes in simulations on multiple programs
- Extensive verification of models performed
  - Developed detailed processes for conversion of the model to C/C++ code
  - Verified performance of the models integrated in the IFS match the performance of the original model as a unit test
  - Regression runs of the full simulation completed to verify performance of the model in the IFS
- Processes updated and tested with latest tool capabilities

Methodical and Thorough Process Used in Development of IFSs using MDE Methods

Performance in Integrated Flight Simulations

Raytheon
Missile Systems

- Wing Actuation System Hardware Model

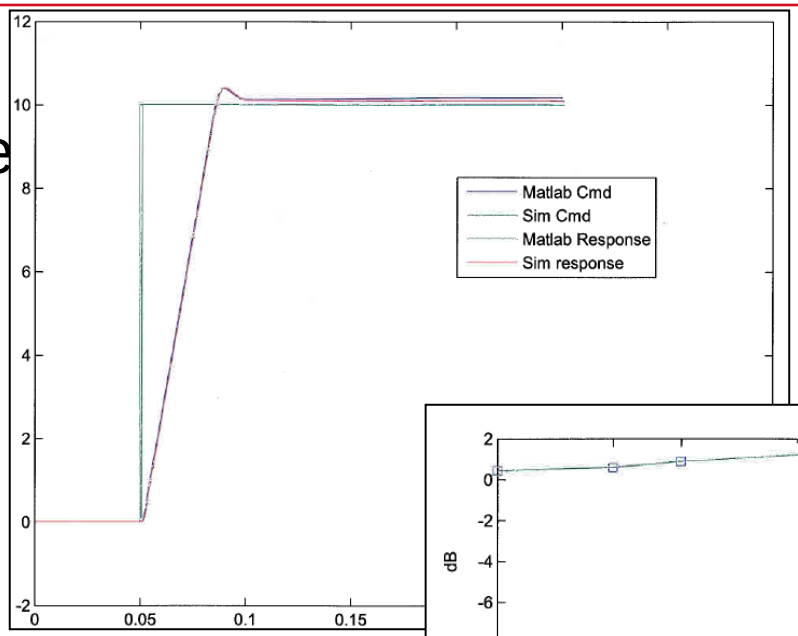


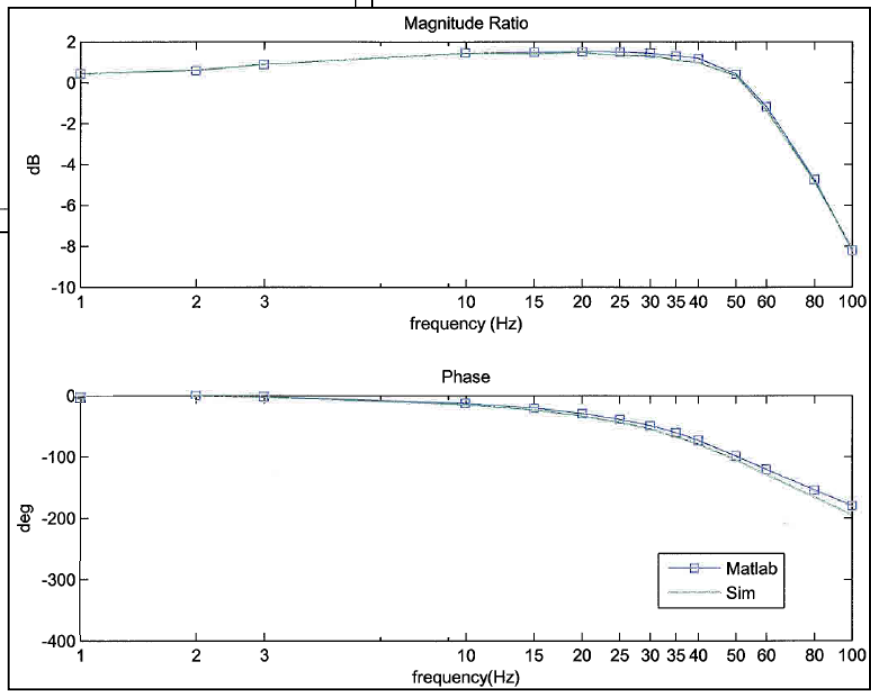
Good Agreement in Time Domain Performance

Performance in Integrated Flight Simulations

Raytheon
Missile Systems

- Control Actuation System Hardware Model
  - Step response
  - Bode Plot





Good Agreement in both Time and Frequency Domains

Common Pitfalls

Raytheon
Missile Systems

- Model Configuration
  - Every model is different, new configurations produce new problems
  - Common model design standards needed for developers to streamline integration into the simulation
- Tool Capabilities
  - As with any tool, user must understand process, model, and MDE tool, not a “push-button” process
  - Common areas to watch
    - Timing – no time shift present
    - Does auto-code accurately represent the system? Auto-code should identically reproduce outputs given identical inputs
- Integration Schemes
  - Internal
    - Continuous – Only available in later releases of Matlab®
    - Discrete – Not always the choice of model developers for representing system
  - External
    - Tie into simulation numerical integration schemes
    - Reduces ability to verify against original model

While MDE tools are useful, care must be taken in model development

Conclusions

Raytheon
Missile Systems

- Raytheon Missile Systems has successfully used MDE processes to incorporate models into IFSs
- Full set of procedures developed to aid personnel cross-program and to train new users
- Procedures verified with multiple models on multiple simulations
- Procedures are updated as new features become available in MDE tools
- Generating code automatically using MDE processes can save significant amounts of time preparing models for incorporation in simulations, and can be completed with confidence