MBD within Airbus-UK Fuel Systems

Opportunities and Experiences
Systems Engineering V-Cycle

Top Level Aircraft Requirements
System Requirements
System Specification
Detail Design
Implementation
System Simulation (A/C -1)
Integration Test (A/C 0)
System Test (SIB)
Unit Test
Airbus Responsibility
Aircraft Test (A/C 1,2,...)

Supplier’s Responsibility

Automated Validation
Automated Testing
Automated Verification
System Design & Implementation
System Design & Integration

© AIRBUS UK LTD. All rights reserved. Confidential and proprietary document.
Model Based Design – Supplier Involvement

AIRBUS

- Design Validation
  - Desktop Simulator
- “Spec In the Loop”
  - Integrated Simulator
- “S/W In the Loop”
  - Desktop Rig
- “H/W in the Loop”
  - Avionics Rig

SUPPLIER

- Sub-System Requirements
  - Equipment Design
  - Component Development

Textual Requirements

© AIRBUS UK LTD. All rights reserved. Confidential and proprietary document.

April 2008 Mathworks AeroDef'08 - MBD within Airbus-UK Fuel Systems
Model Based Design - In Practice

- Rapid Prototyping of Control System Requirements.
  - Normal and Failure Operating Modes
- Simulink/Stateflow Application
  - Platform Independent
  - Exploits DCT
- Control Logic separated from Aircraft Environment
  - System Designers focus on
    - Control Functions
    - HMI
    - Robustness & Validation
  - Specialist Modellers focus on:
    - Aircraft & Environ Simulation
    - GUI/ Panels
    - Auto-Test Capabilities
Model Based Design - In Practice

• Statecharts control behaviour
  ‣ Easier than Enabled/Triggered Subsystems
• Enhanced Validation
  ‣ Statechart representation can be clearer and less ambiguous
  ‣ Increases validation confidence

Fuel System Modelling Environment

Control Function Design

Mathworks AeroDef'08 - MBD within Airbus-UK Fuel Systems
When the model is the requirements, the distinction between “Model Verification” and “Requirements Validation” is somewhat blurred.

If a test fails – is model, the requirement or the test at fault?
Model & Requirement Validation

• Typical Model Development Cycle
  ‣ As model matures, tends towards Requirements Validation

Statistics of Model/Requirement Validation

- Problems Attributed to Model
- Problems Attributed to Requirement

Number of Observations

Time (Months)
Interpretation of ARP4754…

“The complexity of specification written with formalised language raises the need for higher level specification description containing all the requirements implemented in the formalised specification”

- Effectively states that a model is only an implementation of unwritten requirements.

- We need a model and textual requirements in order to sufficiently define and validate a system in compliance with ARP4754
  - Non-Functional Requirements difficult to model
    - Performance / Integrity / Reliability
Model Re-Use – Interface Simulation

- Simulation Platforms have different interfaces
  - Pre-Formatted or Formatted ARINC429/AFDX/CANBUS
  - Includes data for simulation (e.g. Fault Injection)
- Provide Common “Core Model” with specific interfaces
Model Re-Use - Simulation Platforms

• Desktop Simulator
  ‣ Requirements & Environment Model
  ‣ Integrated with Flight Warning & Cockpit Display Models
  ‣ AutoCode using SF Coder & RTW
Model Re-Use - Simulation Platforms

• Aircraft -1
  ▸ Realistic Cockpit Mock-Up
  ▸ Simulated Avionics
  ▸ Interfaces Identical to Full Flight Simulator
Model Re-Use - Simulation Platforms

• Aircraft Zero (Iron Bird)
  ‣ Cockpit Avionics & Displays
  ‣ Real or Simulated Avionics Equipment (Interchangeable)
  ‣ Simulated Environment
Model Re-Use - Simulation Platforms

• Full Flight Simulator
  ‣ Fully Simulated Systems and Environment
  ‣ Single model for all platforms
    – Interfaces pre-configured for each platform
Multi Team Model Design Process

Design Teams
- Engine Feed
- Ground Ops
- Transfer
- Indication

Config Controller

Fuel System C&I Model (Baseline)

Changed Items

To Supplier
- Fuel SSRD

To Integrators
- Simulator Code

Simulator Model

To Supplier
New Developments - Formal Methods

• MBD is not “Formal” in the mathematical sense
  ▸ Once created possible to apply formal methods
• Proof Technology – Design Verifier
  ▸ Mathematical analysis of the Model
    – without traversing all possible scenarios
    – complete in a mathematical sense
      • correct and desired behaviour
      • wrong and not desired behaviour
• Some restrictions may hinder progress
  ▸ E.g. Non-Virtual Buses, Stateflow Structures
  ▸ Model may need changing to make it Validatable
    – May alter the intent of the requirement
New Developments - Static Analysis

• Static Code Analysis
  ‣ Ability to “prove” correctness of code
    - Out of bound values
    - Divide by zero
    - Infinite Loops
    - Overflow/Underflow
    - Unreachable code/modules
    - Square Root negative numbers

• Polyspace Model-Link
  ‣ Auto Generate Code to Analyse Model
Lessons Learnt - Model Based Design

• Model build process can reveal anomalies/ambiguities
  ‣ Validation for free
    – Identify Assumptions separately from requirements
    – Identify Executable Implementation from Requirements

• Validation Testing
  ‣ A test that is more complex than that being tested is probably wrong
  ‣ Easy to be caught in the trap of “Test for Success”
    – Testing for intentional, but not unintentional behaviour
    – Project managers demand simple progress metrics

• Model Architecture
  ‣ Separate Requirements Model from Environment Model
  ‣ Separate real interfaces from simulator
Lessons Learnt – System Design

• System Designers focus on Designing the System
  ‣ The System Model is the System Requirements
    – Extra functionality required to exercise the model are not requirements

• Discontinuity between Design and Implementation
  ‣ Detailed Models required for Integration Simulators
    – Required before availability of equipment
    – Need to create models of potential implementation

• Easy for Designers can be Difficult for Simulators
  ‣ Matlab Function Blocks
  ‣ M-File S-Functions
  ‣ Test Harnesses
    – Can break the automatic code generators
Lessons Learnt - Migration

• Aircraft Life Cycle w.r.t. COTS
  ‣ A/C measured in Decades – COTs measured in Months.
    ‣ Tool versions will become obsolete – so must plan it in from start

• Cost of upgrading
  ‣ Installation, Training, Hardware
  ‣ Rework obsolete features, Model regression testing & re-validation

• Benefit (Cost of not upgrading)
  ‣ Bugs
  ‣ Utilization of new features
Any Questions?
© AIRBUS UK LTD. All rights reserved. Confidential and proprietary document.

This document and all information contained herein is the sole property of AIRBUS UK LTD. No intellectual property rights are granted by the delivery of this document or the disclosure of its content. This document shall not be reproduced or disclosed to a third party without the express written consent of AIRBUS UK LTD. This document and its content shall not be used for any purpose other than that for which it is supplied.

The statements made herein do not constitute an offer. They are based on the mentioned assumptions and are expressed in good faith. Where the supporting grounds for these statements are not shown, AIRBUS UK LTD will be pleased to explain the basis thereof.