Model-Based Systems Engineering without SysML

John R. Palmer
Michael E. Crow
Ronald S. Carson, PhD

The Boeing Company

Presented to National Defense Industrial Association
Systems Engineering Conference, October 24-27, 2011

Motivation

- **There appears to be a general assumption that SysML and UML are synonymous with Model-Based System Engineering**
  - OMG - Certified Systems Modeling Professional
    - http://www.omg.org/ocsmp/exam-info.htm
  - OMG - Ontology Working Group
    - "SysML plays a benchmark role in the MBSE ontology activity"
      - http://www.omgwiki.org/MBSE/doku.php?id=mbse:background_material_for_mbse_ontology_development
  - Interface Communication Modeling Language based on UML
    - http://www.insidegnss.com/auto/IGM_janfeb11-Gianni.pdf

This presentation aims to offer an alternative

What is MBSE?

- INCOSE Vision 2020
  - "Model-based systems engineering (MBSE) is
    - "the formalized application of modeling to support system requirements, design, analysis, verification and validation activities
    - "beginning in the conceptual design phase and continuing throughout development and later life cycle phases.
  - "MBSE is part of a long-term trend toward model-centric approaches adopted by other engineering disciplines, including mechanical, electrical and software.
  - "In particular, MBSE is expected to replace the document-centric approach that has been practiced by systems engineers in the past and influence the future practice of systems engineering by being fully integrated into the definition of systems engineering processes."

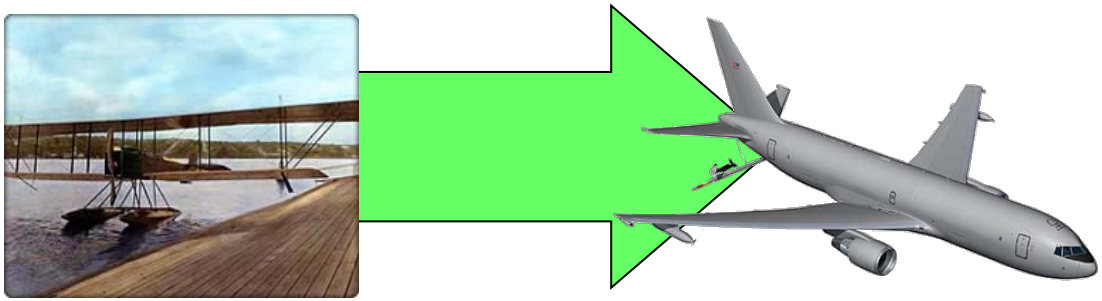
- Reference: "International Council on Systems Engineering, Systems Engineering Vision 2020," INCOSE-TP-2004-004-02, Version/Revision: 2.03, Dated September 2007

What is a Model? Why Model?

- **What is a model?**
  - Model: N. “a simplified representation of a system or phenomenon, as in the sciences or economics, with any hypotheses required to describe the system or explain the phenomenon, often mathematically.”
  - To Model: V. “to simulate (a process, concept, or the operation of a system), commonly with the aid of a computer.”
  - For MBSE, the model often precedes the reality
- **Why model?**
  - What does it enable that you otherwise wouldn’t have without it?
    - Identify missing concepts or objects
    - Identify emergent capability from data (planned or surprise)
    - Support additional analytical capability (e.g., fault analysis)

Find problems earlier → Faster, better, cheaper

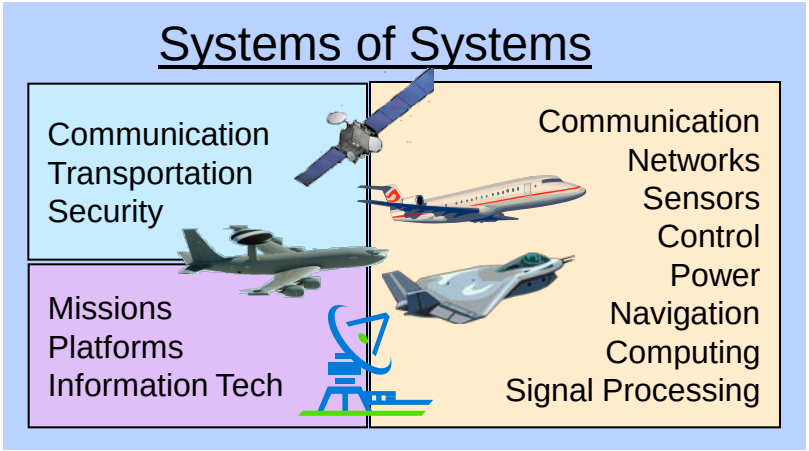
Evolution of Aerospace Architectures

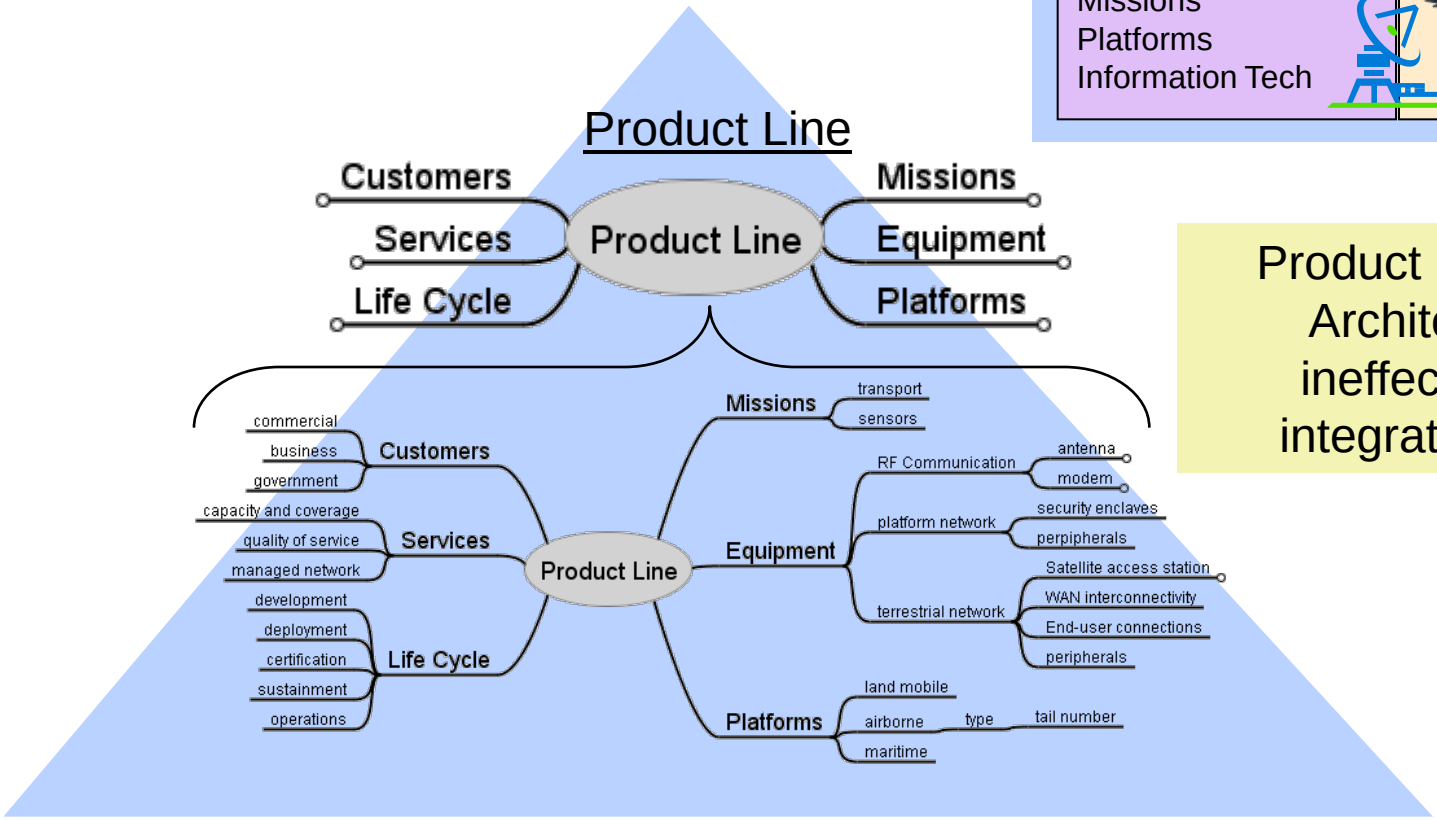


	Federated	Integrated, Distributed
Functional interactions	“Brick wall” systems	Shared common resources; Many interacting systems
Interface Definition	Single designer, functional cohesion, loose coupling	Many developers Performance demands drive interface complexity
Integration effort	Simple	Intricate functionality
Part and wires count	High	Lower
Failure behavior	Failures more visible	Complicated integrated behavior Failures more “opaque”

Integrated, Distributed System quality can be challenging to manage without integrated model

Challenging MBSE Use Cases





Product Line and SoS Architectures are ineffective without integrated modeling

MBSE Implementation Requires Many Considerations

- **Program buy-in - “If not in the model, it’s not in the project”**
- **Doing System Engineering in the MBSE paradigm**
  - Analytical validation of the architecture model
  - Analytical verification of the system architecture
  - Customer artifacts generated directly from the model
  - Detailed design features applied directly from the model
- **System life cycle**
  - Product, test, production, and sustainment
  - Program effectiveness and suitability
    - Early and continuous accounting for quality attributes
- **Program transitions to MBSE**
  - Representation of the information model
  - MBSE environment and related training
  - Phased capability deployment - data conversion and generation

Evaluation Criteria for Successful MBSE

Effectiveness
- Model completeness and correctness
- Modeling productivity (initial and changes)

Model Requirements
- Product line and life cycle characteristics
- Integration complexity
- Stakeholder artifacts

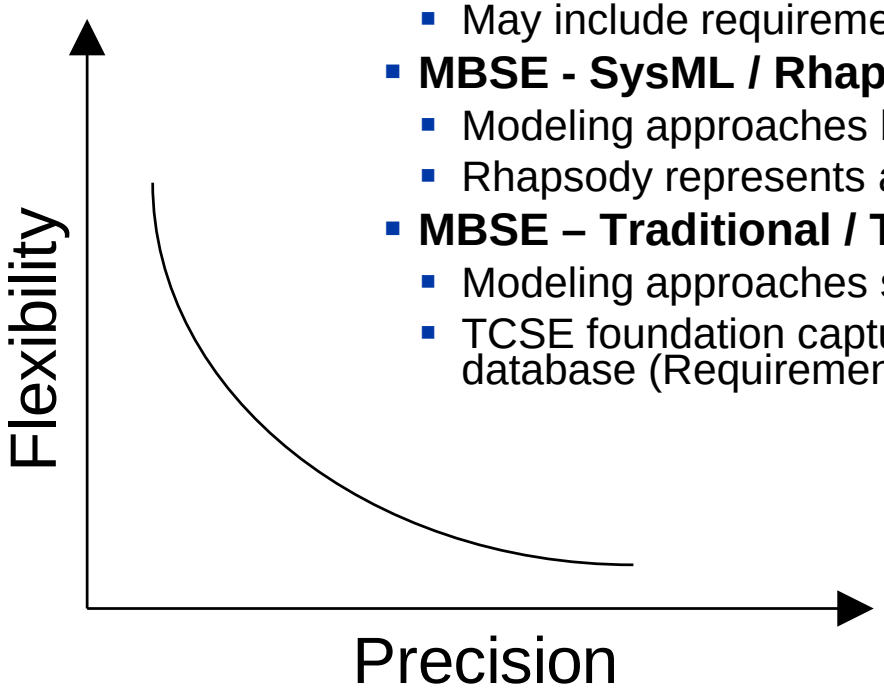
Suitability
- Consistent with organization processes and practitioner expertise
- Computing/data/tool infrastructure
- Team geographical distribution

MBSE Approach Evaluation Criteria		
Model Precision	Process Compatibility	Size Limits
Model Semantics	Skill Compatibility (CMMI)	Data Queries
Business Rules	Development Environment	Configuration & Change Management

Evaluation criteria based on effectiveness, model requirements, and suitability considerations

Modeling Processes and Tools Options

- **Traditional**
  - Modeling approaches such as FFBD, IDEF0, IBD
  - Modeling done without the aid of tool-supported integration
  - May include requirements database a la DOORS
- **MBSE - SysML / Rhapsody**
  - Modeling approaches based on SysML (UML foundation)
  - Rhapsody represents an industry leading SysML implementation
- **MBSE – Traditional / Teamcenter System Engineering (TCSE)**
  - Modeling approaches such as FFBD, IDEF0, IBD
  - TCSE foundation captures architecture elements in a single database (Requirements, Functions, Logical, etc)



Modeling alternatives represent a trade of flexibility and precision

Modeling Alternative Evaluation (1/2)

	Process/Tool		
Criterion	Traditional	SysML / Rhapsody	Non-SysML / TcSE
Model Precision	• Low - RM tool, independent diagrams	• High	• High - depends on meta-model
Model Semantics	• Loosely defined set of customizable semantics	• Integrated, complex but defined set of extensible semantics	• Customizable semantics - depends on meta-model
Business Rules	• Process-enforced	• Tool-aided	• Tool-aided
Process Compatibility	• Consistent with IEEE 1220 language	• Less Consistent with IEEE1220 language	• Consistent with IEEE 1220 language
Skill Compatibility (CMMI)	• Consistent with engineering skills	• Most SEs not familiar with SysML • More difficult training	• Consistent with engineering skill base
Development Environment	• No integrated data • Engineers tend to work independently	• Integrated Model, not integrated to other data • Single file accessible to single user • Jazz enables access to different tools/data	• All SE data integrated in one database • Multi-user distributed environment • Web-based client-server architecture

Modeling Alternative Evaluation (2/2)

	Process/Tool		
Criterion	**Traditional**	**SysML / Rhapsody**	**Non-SysML / TcSE**
Size Limits	• Management problem with large data sets	• Models limited in size by local memory	• Effectively unlimited model size
Data Queries	• Not possible across architecture views	• Complex queries require coding • Segmentation limits	• Query Wizard supports complex queries
Configuration and Change Management	• Manual development and integration • Document-based • Copy for variants	• Model CM via Clearcase; other data handled separately • Accommodates check-out/-in, branch, merge • Delayed synchronization	• All SE data managed in single database • Data overwrite is possible without check-out or process control • Limited versioning and variants OOTB • Accommodates branching in model • Immediate synchronization

Key Features to Non-SysML / TcSE Approach

- **Integrated data environment**
  - → Ease of data coordination among design disciplines
  - → Common change management capability
  - → Immediate knowledge of the state of the system design
  - → Design artifacts can be generated automatically
    - – Document trees, specifications and interface control documents, verification reports, …
    - – Architecture development metrics
- **Multi-user environment**
  - → Engineering disciplines work concurrently from single authoritative data source
  - → Supports very large system models (millions of objects)
  - → Supports geographically dispersed organization involvement
- **Immediate synchronization**
  - → The latest system engineering data is available to everyone
- **Robust query engine allows rapid assessment of integrated database**
  - → Completeness of requirements, functions and other system characteristics
  - → Traceability among these elements
  - → Analytical validation of system design

Streamlined system engineering and integration, higher design efficiency, and fewer errors in complex design efforts

Conclusions

- **Criteria leading to non-SysML alternative selection**
  - Very large projects with multi-user access → TcSE
  - Global search and consistency checking → TcSE
  - IEEE1220-based process definition and training → TcSE
- **Issues to be managed**
  - Semantics must be defined and controlled to align with organization processes
  - Multi-user effects on data and configuration control
- **Other representation formats are not precluded**
  - SysML and DoDAF artifacts can be prepared from the same data
  - Requires corresponding metamodel definition
- **Successes to date – realized integration analytical capability**

MBSE can be effective without SysML