JPL Community View on Challenges and Rewards of MBSE

Bjorn Cole, PhD
Jet Propulsion Laboratory
California Institute of Technology
Bjorn.cole@jpl.nasa.gov
Agenda

• Rewards:
  – Doing the same job better
  – Bending the complication-effort curve
  – New capabilities
  – View interdependence with a model

• Challenges:
  – Model management (establishing baselines, checks before “publishing,” library building and sharing, fighting “model rot”, organization) (4 minutes)
  – A maturing standard with some deficiencies (tool vendors need flex and adaptability; models may get hammered by required refactoring) (3 minutes)
  – Tooling interfaces (need to get unified and open to avoid many point-to-point connections) (4 minutes)
  – “Soft support” (impacts to processes, training, etc.) (2 minutes)
  – New resource requirements (dedicated modelers and tools) (2 minutes)
  – Model applicability / tailoring to an effort (2 minutes)

• Q & A (5 minutes)
Who Do I Represent Today?

- Technically, myself
- This is no official pronouncement of the “JPL message” to INCOSE or anyone else in the audience
- But the message is from an **MBSE community member** that is **contextualized** by experiences seen (directly and second-hand) at JPL through the Modeling Early Adopters group and our Integrated-Model Centric Engineering (IMCE) initiative; also selected concept / flight project teams
- We have shown the **feasibility** of MBSE (in pieces) in the work I have seen; we are really starting to fight the battles of **practicality** and **usability**
- My experience does not come without late nights and some light scarring (of course, my life pre-modeling had a lot of this too...)
One Final Caveat

- MBSE includes more than SysML
  - We can unify multiple languages / tools / methods by looking at the viewpoint / view perspective (elaborated later)
  - We have multiple engineers looking at combining the best of a more formal language (OWL) and the graphical expressions/views of SysML while easing its internal contradictions and quirks
  - Starting to integrate analytical models
  - Business Process modeling also an important standard for workflow and processes for using a given system

- So now that we are scoped …
Rewards of MBSE
3.2.4. Information Screening

Text:
Furnace shall provide access to data and models only to authorized requesters.

Rationale:
(from Authorization in the Loop):
The Foundry deals with a large quantity of information that is sensitive in a variety of contexts.

1 Jackson, M., “Dynamic Gate Product and Artifact Generation from System Models,” IEEE Aerospace Conference; 5-12 Mar. 2011; Big Sky, MT; United States

New Capabilities: Tradespace Generation

1 Cole, B., “Analyses Made to Order: Using Transformation to Rapidly Configure a Multidisciplinary Environment”, IEEE Aerospace Conference; 5-12 Mar. 2011; Big Sky, MT; United States
Analyzing Components in Use
Same Job Better (1)

- Formal rules can help tremendously with systems engineering rigor and hygiene
- IMCE leads the way with automated checking of their own modeling products
- Project pilots are demonstrating formal rule usage¹
  - Is the definition complete? (e.g., is everything typed?)
  - Is the definition consistent? (do your types conflict?)
  - Is uniqueness satisfied?
  - Do abstract interfaces eventually get implemented?
  - What are topology metrics for the system (e.g., network latency?)

Same Job Better (2)

- Consider the DARTS spacecraft:\n  - Straightforward architecture: System uses filtered software estimation of position/velocity unless measurement outside allowable limits
  - Measurement provided by visual system alone, or by visual + GPS?
  - Software implementation diverged from architectural understanding (visual only) → frequent resets → catastrophic navigation error

- Mars Reconnaissance Orbiter:\n  - Again, straightforward architecture: Control logic applies a “Keep Out Zone” to solar wings with margin to prevent impact to the spacecraft
  - KOZ definition eroded over time and translation until wings were allowed to penetrate zone with more velocity than could be arrested at max deceleration; impact on spacecraft
  - Software implementation diverged from architectural understanding

Bending the Complication Curve

- Multiple views reduce dimensionality for understanding
- Challenge of consistency makes SE’s bookkeepers in an $n^2$ problem
- Alternative is the unified model where rigor defends view consistency
View Interdependence

- UML / SysML originated as a coalescence of views with (mostly) understood common semantics
- Instinct for working with engineers good to have work done from views
- Thrusts in formality\(^1,\)^\(^2\) to more deeply assure consistency and consequences in other views is key


What Does This Mean Practically?

• As systems get more complicated, the adversary is less physics and more our understanding of what we’ve actually done

• Two big thrusts: productivity (in certain areas) and trust in our work

• Productivity is in more trades faster, retaining of knowledge (through trust in applicability), and more intelligent heritage

• Trust is in formal guarantees
  – Facts stated in one description faithfully translated to others
  – As in control theory, relational database design, etc., theorems can assure easy-to-verify properties lead to harder-to-verify ones (stability, completeness, freedom from classes of errors)
Challenges of MBSE
Model Management: Compiling a System

- Layers of formality and processing support around system modeling imply that a system can one day be “compiled” from an evolving model
- Models share a good deal of attributes and requirements of software
  - Version control
  - Dependency resolution
  - Data integrity
  - Strong desire for branching, merging, and roll-back
  - Many opportunities in reuse
Dependencies and Referential Integrity

• Different groups will want control over different parts of the model
• Import libraries and fragments of other models
• Single database with consistency assurances v. replication and distributed entry
• How do you control change propagation?
  – We have a desire for high agility → propagate it all
  – But don’t want to wake up day before a review to see everything changed
  – System-wide investigations of alternatives to be coordinated by the model require model-wide changes; a separate branch?
Libraries When and How

• A challenge is to balance desire for reuse and knowing when reuse is ready
  – A library made too early is unstable and likely incomplete
  – Waiting too long to make a library makes efforts redundant
  – If individual tasks undertake to build a library, can easily lose visibility about who is depending on it and implicitly intertwined schedules
  – When do you know “best practices” are best?

• Library releases (and accompanying scripts, rules, etc.) need to have version loading controlled
Working the Standard

- SysML is an evolving standard with version updates containing major improvements
  - E.g., SysML 1.3 made major changes to ports and flows for the better
  - Perfect has not been allowed to be enemy of the good
- As with any software, returns to a classic dilemma: compromise improvements for backward compatibility or leave legacy users behind
- If refactors are too labor intensive, will degrade the MBSE experience
  - Flows and ports pretty straightforward
  - Consider a need to refactor all behavioral parts of a SysML model if something like “ontological behavior” is implemented
- Think of a major project lasting for ten years that adopts MBSE from the beginning
Rules May Help

- OMG is starting to develop versions of standards with machine-readable updates
- Non-competitive transformation rules could help all vendors alleviate transition issues for clients
- An opportunity for assured migration during standards evolution can reduce the risk and improve the experience of working with the standard
Tooling Interfaces

• The drive is to analytical integration
• How to avoid many point-to-point connections?
• Consider the desire to connect SysML tools and …
  – DOORS
  – Satellite Toolkit
  – Math solvers
  – Modelica tools
  – Mechanical CAD tools
  – Electrical CAD tools
  – Operational research tools
  – Campaign simulation tools
  – Process tools
Some Approaches

- Projection between models of elements for interconnection in native domain (e.g., project relevant properties of CAD entities into SysML for connection to other properties)
- Semantic coordination (e.g., have a definition for time and precedence all tools can leverage; STEP)
  - Geometry
  - Time
  - Space
  - Information Sharing; Cause and Effect
- Connect relevant parameters
- Treat all models as views on some supermodel
- There are tool vendors investigating all approaches
The Softer Side: Training (1)

- Model-Based Systems Engineering is interesting, large topic
- When I practice, I exercise skills learned in:
  - Programming: Model Queries, Transformations
  - Mathematics: (Basic) Set Theory, Graph Theory, Geometry, Differential Equations, Algebra, …
  - Computer Science: Formal logic, rules
  - Domain engineering: Aerospace training to know when to apply what from the above
- Lots of training to be an expert developer
- How to transition to a smaller set to be expert user?
The Softer Side: Training (2)

- A lot of my challenges in relating MBSE, especially with SysML, is relating computer science methods and object-orientation:
  - Typing / Classification
  - Inheritance as specialization v. inheritance as importation of properties and methods (mix-ins and aspects?)
  - Levels of abstraction and instantiation
  - Rules and traverses on the model and how to organize it for their efficiency / feasibility
  - Formally describing “what everyone knows”

- Note that none of the above gets to the number-crunching many of us are trained to do!
  - Moreover, just going out and prototyping
The Softer Side: Training (3)

- The fine line: “How can I get the benefits without a lot of the training?”
  - At one level, the same question as “how do I engineer without learning math?”
  - On another, have to be careful to not let this become “how do I engineer without a Ph.D. in math?”
  - Can help bend the curve with customized interfaces, limited perspectives, and carefully crafted views
- Managerial side: how much do I need of each of above?
- The challenge: Match traditional domain approach in view but still be clear about what is in model
Added Resource Needs? (1)

- Increasing specialization applies pressure for increased staffing
- Can all SE’s become MBSE’s?
- Are all mechanical engineers now conversant with some kind of CAD?
- To be cost-effective, the MBSEEngineer must either reduce more cost than he or she incurs, generate enough profit, or assure enough value to be offset
Added Resource Needs? (2)

• Right now, we are in the pinch-point
  – Challenging to find MBSE practitioners that can substitute in for more traditional systems engineers
    ▪ Lower experience
    ▪ May be less domain-oriented than others
  – Reward is in the future, cost is in the present

• But is the leap of faith really that great?
  – Quality has been seen to reduce lifecycle costs and bring happier customers
    ▪ Think Toyota, Honda
  – The pain is often in the interfaces and assumptions on operational conditions; reducing pain there can be a big help
Another Practical Summary

- Models of complicated systems are themselves complicated artifacts (but not nearly as complicated...)
- MBSE isn’t first time we’ve modeled systems, but it is the best real stab so far at modeling in a consistent way
- Not as many new skills required but a new combination of skills required
- The challenges are a blend of challenges from dealing with handling large intellectual products (MBSE’s original problem it meant to attack) and handling software development tasks
Cost/Benefit
Total Summary

• Benefits are not imagined; we have seen them begin to emerge
  – Coordination and greatly enhanced traceability; connecting the dots is baked in if you do it right
  – Automated verification
  – Automated documents!
  – Generating many cases from rules
  – Asking the model interesting questions

• Challenges are not imagined either; multiple experiences of growing pains are real too
  – Dealing with model reference issues
  – Premature attempts at building libraries
  – Hard to scale up trained workforce after natural adopters have been converted
  – Questions of how much to integrate and when