Multi-Paradigm Modeling and Simulation, a Systems Engineering perspective

Jerome Hugues

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM22-0544

BLUF: MPM or MBSE? Both!
(Let's start with the conclusion)

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."* (INCOSE 2007)

"Multi-Paradigm Modeling (MPM) offers a foundational framework for gluing the several disciplines together in a consistent way. [..] MPM offers processes and tools that can combine, couple, and integrate each of the views that compose a system." [MPM4CPS CfP]

Two theses

- *Claim#1: one cannot do MBSE without MPM*
- *Claim#2: one should not do MPM without MBSE*

Outline

1. Context
2. About AADL
3. *Claim#1: one cannot do MBSE without MPM*
4. *Claim#2: one should not apply MPM without MBSE*

Overarching objectives – or what I am striving for

But is it MBSE or MPM ?

[….] assists engineers (mainly software, but also CPS and systems engineers) with models to

Concepts

Organize stakeholders needs and elicit requirements

Capture system elements – design, reverse engineering or COTS

- Interfaces, components internals (static and behavioral), and
- a system architecture built from those: deployment, (re-)configuration

Apply analytical frameworks to assess model's "compliance to some objectives"

- Syntactic, conformance to guidelines, patterns "well-formedness"
- Quality of system, w.r.t. performance, safety, security, behavior metrics
- Behavioral correctness, e.g. model checking, simulation, proofs

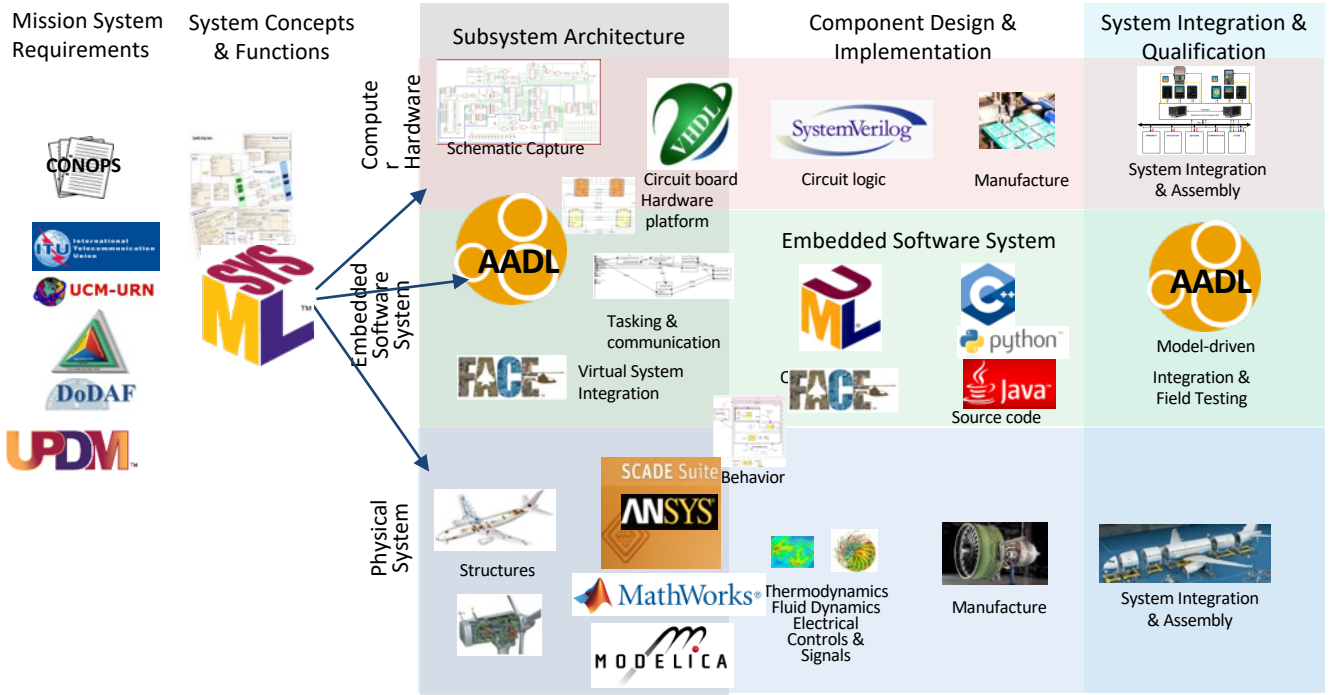
Models as processable artifacts to guide the software engineering process

System

- "Modeling is the new programming"

Provide more insights than CAD or code-only solution through relevant abstractions and automation

MPM landscape – science removed for the faint of heart



ISO15288 – Processes for Systems Engineering

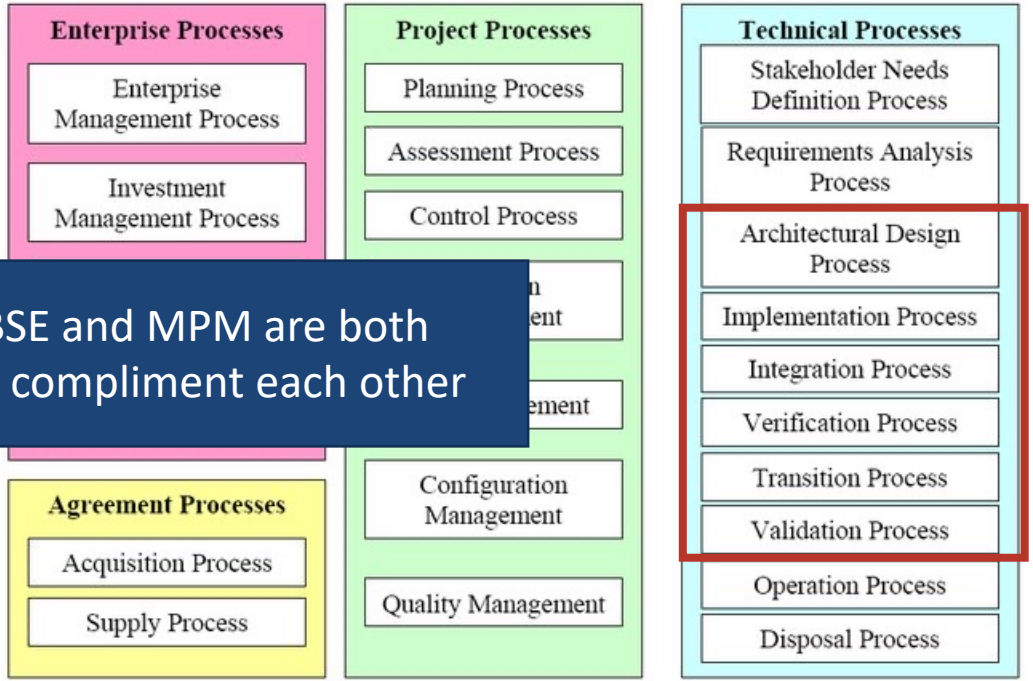
A process is a set of interrelated or interacting activities which transforms inputs into outputs (ISO 9000:2005)

- Process are organized so that they contribute to a specific goal. They are structured by an *argument* that justifies the existence

Hence, MBSE and MPM are both relevant and compliment each other

MPM provides solutions only for a subset .. But interconnected with all other process

MBSE covers all, but lack many advanced V&V or simulation support



The system life cycle processes Source: ISO/IEC 15288, Figure D.8
ISO 15288 hass **23+ processes** which have **123 outcomes** derived from **403 activities**.

Outline

1. Context
2. About AADL
3. Claim#1: one cannot do MBSE without MPM
4. Claim#2: one should not apply MPM without MBSE

The Safety-Critical Embedded Software System Challenge

Problem:

- Software increasingly dominates safety and mission-critical system development
- Issues discovered long after they are created

Goal:

Early discovery of system-level issues through virtual integration and incremental analytical assurance

Solution:

- **Language** standardized via SAE International & matured into practice through pilot projects & industry initiatives
- **Tooling** available under open source license continually enhances analysis, verification, and generation capabilities
- **Expertise** in Modeling Safety-Critical Embedded Systems

Model-Based Engineering for Cyber-Physical Systems @SEI

Create the best design that holds up over time as the system evolves. + Test the design without having to write any code. = Build a single model to assess hardware and embedded software before the system is built.

SAE AADL / ACVIP

Standardized language and process for the engineering safety-critical systems.

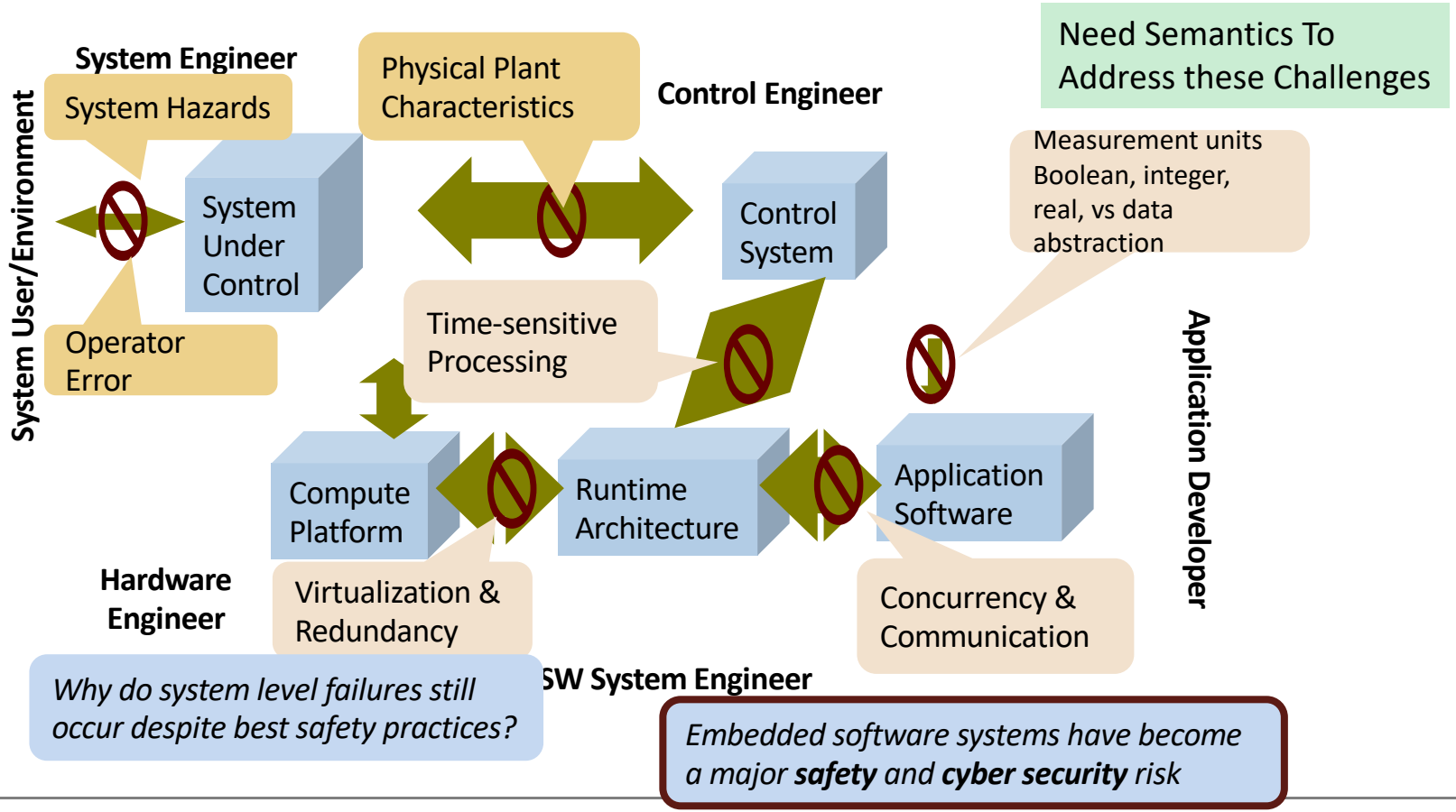
OSATE

Open Source AADL toolset for performing verification and validation (V&V).

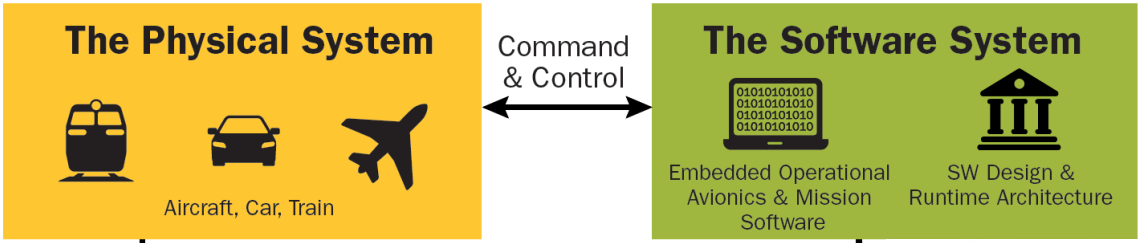
DoD Transitioning

Maturity increased through pilot projects and trainings.

Interleaved problems – why we need MPM-like techniques



Architecture Analysis & Design Language (AADL) Standard Targets Embedded Software Svstems



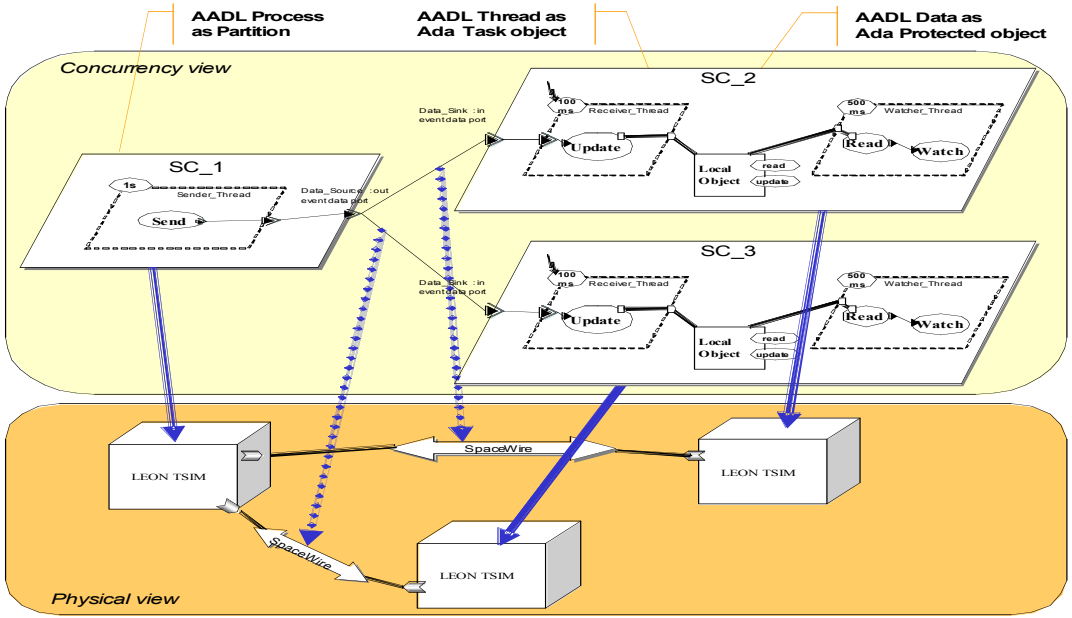
SAE International
AS 5506 Standard Suite

Standards provide long-term industry-wide solutions to support multi-organization model-based engineering

In 2008 Aerospace industry initiative chose AADL over SysML and other notations as it specifically addresses embedded software systems

AADL captures mission and safety critical embedded software system architectures in virtually integrated analyzable models to discover system level problems early and construct implementations from verified models

AADL



General AADL tool: OSATE

Requirements verifications: ALISA

Middleware synthesis : Ada, C (POSIX, ARINC653), RTOS: Ocarina

Scheduling: Gateways to Cheddar, MAST

Error modeling, FTA, FMEA: OSATE

Model checking: Petri Nets (Time and CPN), LNT

Security Analysis, CI/CD, […]

AADL goal is to provide “ground truth” for describing safety-critical systems

SAE International AADL Standard Suite (AS-5506 series)

Core AADL language standard [V1 2004, V2 2012, V2.2 2017, V2.3 2022]

- Focused on *embedded software system modeling, analysis, and generation*

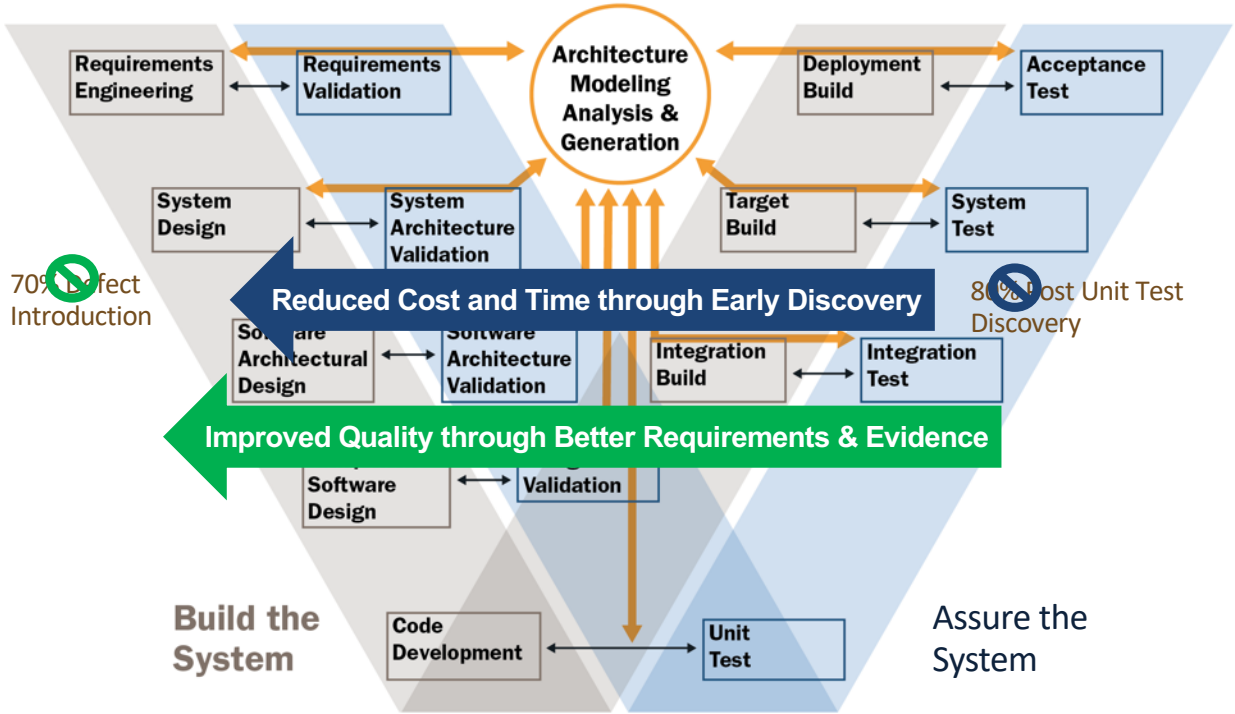
Standardized AADL Annex Extensions
- Error Model language for safety, reliability, security analysis [2006, 2015]
- ARINC653 extension for partitioned architectures [2011, 2015]
- Behavior Specification Language for modes and interaction behavior [2011, 2017]
- Data Modeling extension for interfacing with data models (UML, ASN.1, …) [2011]
- AADL Runtime System & Code Generation [2006, 2015]
- FACE Annex [2019]

- Evidence produced as a result of automated tool-supported analysis
  - Performance analysis: worst-case response time, schedulability of the system
  - Safety analysis: computing fault trees, probability of reaching an unsafe state
  - Automated model review: conformance to modeling guidelines
  - Code generation: generating “correct-by-construction” software

TQL-5, i.e. verification tools. Output :- evidence, part of some argument

TQL-1 tools, i.e. output is part of final system

Improved Cost, Time and Quality



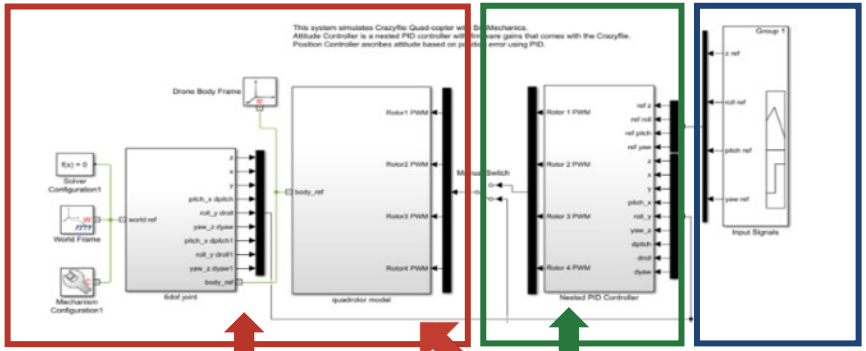
Outline

1. Context
2. About AADL
3. Claim#1: one cannot do MBSE without MPM
4. Claim#2: one should not apply MPM without MBSE

TwinOps problem space: CPS Integration and Testing

See [MPM4CPS'20] for details

High-Level Architecture

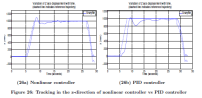


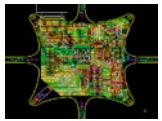
TwinOps: leverage other source of truth (e.g., CAD, Physics) to improve SW V&V

⇒ Use precise models instead of (naïve) abstractions for improved SW V&V

⇒ Combine domains, including SysEng

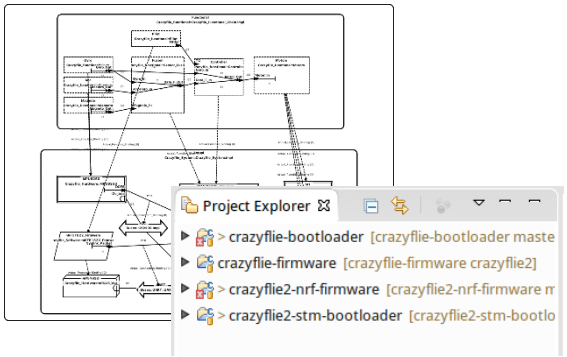
Implementation Space





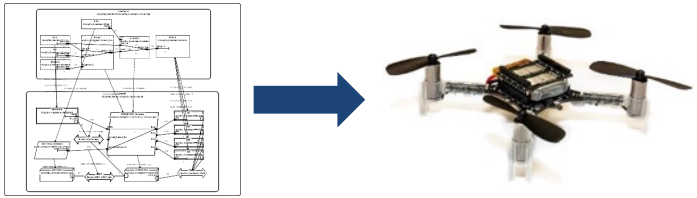
+

SW/HW Architecture



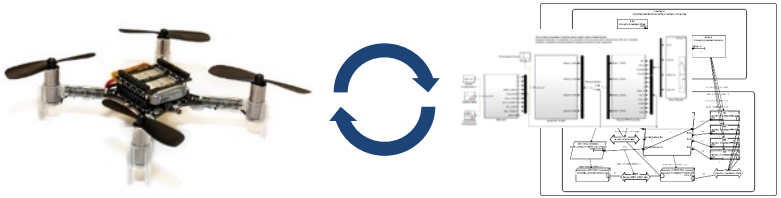
Technology Focus: Models and Code Generation

One can ***generate code*** from models ready to be embedded in the system (e.g., AADL to C) and get insights from the system to refine the model metrics.

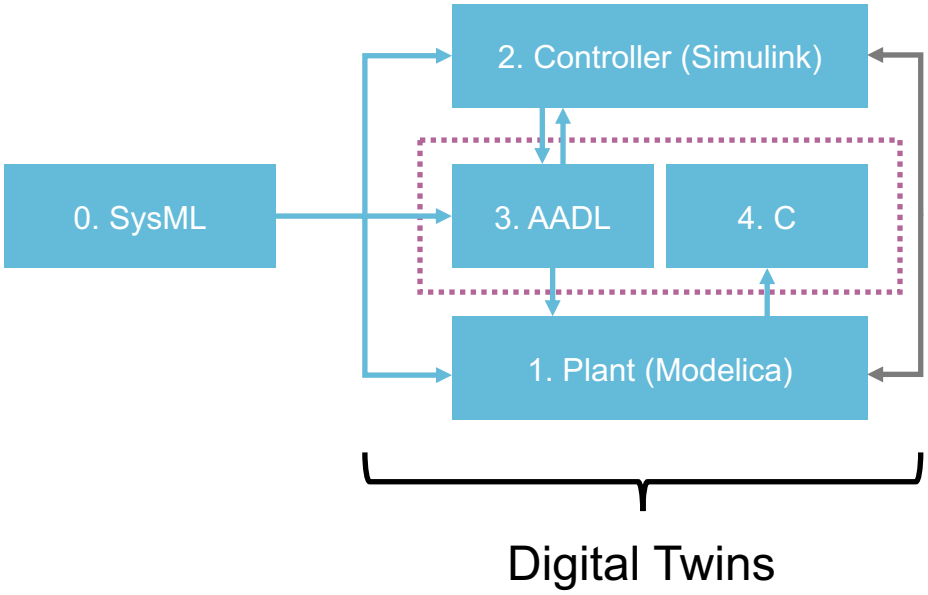


One can ***simulate models*** and generate simulation code as a mock-up of some system parts.

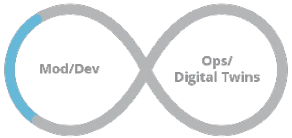
One can build ***Digital Twins***, that compare actual system and its digital simulated doppelganger.



MBSE and MPM, => MBSE perspective

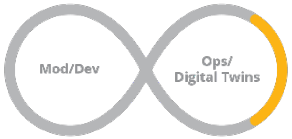


1-2-3-4: “mega-modeling” V&V
- 1-2: HLR validation
- 2-(3+4): validation of LLR
  1+(3+4): virtual integration



Digital Twins of UAV vs. UAV flying: validation of Modelica model, efficiency of the controller (overshoot verification) and timing verification of software.

From ModDevOps to TwinOps



SCM

Mod2Code

Cloud Deployment

SCM

Mod2Simu

Digital Twins

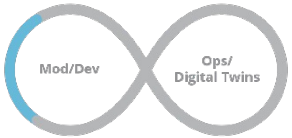
Modeling Process

Outline

1. Context
2. About AADL
3. Claim#1: one cannot do MBSE without MPM
4. Claim#2: one should not apply MPM without MBSE

MBSE and MPM, => MPM perspective

1-2-3-4: “mega-modeling” V&V
- 1-2: HLR validation
- 2-(3+4): validation of LLR



How to organize models? Matryoshka principles
- AADL extends/refines the SysML model block diagrams
- Simulink implements requirements for a controller
- Modelica implements operational context

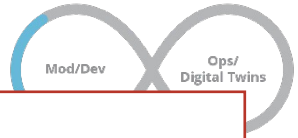
=> notional process encoded, “$model_1$ + operation => $model_2$”

Caveat: Liskov principle: a model refining another model should preserve some properties (structural, behavioral)

MPM must build on Systems Engineering and MBSE to define
- Each process involved
- Goal of each step within a process

MBSE and MPM, => MPM perspective

1-2-3-4: “mega-modeling” V&V



How to “run” models?

- AADL orchestrates software execution on target
- FMI to do co-simulation AADL/Simulink/Modelica

AADL architecture acts as a master algorithm for simulation

- AADL orchestrates the execution of simulation models of devices (models of the environment) and software (e.g. SCADE, Simulink, etc)
- Underlying assumptions on clocks, sampling time dictated by the schedule constraints set by the architecture

What about system-level simulation?

- WiP: DEVS to provide semantics of AADL itself
- MPM should embrace all aspects of a CPS, need faithful characterization of the cyber component
- How to provide “science” in the way we assemble those models?

Conclusion

Just opening the box: MBSE and MPM are two fruitful research topics

“engineering” dimension: support the construction of large complex systems

- With enough confidence models are useful!
- Interoperability solved through DEVS/FMI, meta-modeling, and model transformations

“scientific” dimension TBD to achieve confidence:

- Heterogeneous models can be weaved and
- The resulting (coupled) model has a valid semantics, i.e. a valid interpretation

Conflicting issues: scalability/fidelity, i.e. finding the right abstraction for the right property