Model-Based Systems Engineering (MBSE): An Architectural Perspective

Dio de Niz, Sam Procter, and Jérôme 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.

DM22-0949

Outline

Model Based Engineering at the SEI

The Safety Critical Embedded Software System Challenge

Problem:

- Software increasingly dominates safety and mission critical system development cost
- 80% of issues discovered post unit test

Context:

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

Solution:

- Technology based on SAE International standard matured into practice through pilot projects and industry initiatives
- Open source research prototyping platform continually enhances analysis, verification, and generation capabilities
- Direct alignment with DoD Digital Engineering Strategy

Model-Based Engineering for Cyber-Physical Systems

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.

Before You Even Write a Line of Code…

AADL allows you to design the entire system and see where the problems may occur. Then you can change the design of the system to eliminate those errors.

Being able to perform a virtual integration of the software, hardware, and system is the key to identifying problems early – and changing the design to ensure those problems will not occur.

About AADL

- SAE Avionics AADL standard adopted in 2004
- Focused on embedded software system modeling, analysis, and generation
- Strongly typed language with well-defined semantics
- Used for critical systems in domains such as avionics, aerospace, medical, nuclear, automotive, and robotics

Outline

Model Based Engineering at the SEI

Model-Based Systems Engineering and the impact of Architecture

Systems Engineering 101

"Systems Engineering is a transdisciplinary and integrative approach to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods." [INCOSE]

Where do we start?

[SEBOK]: INCOSE Systems Engineering Body of Knowledge

[NASA]: NASA Systems Engineering Handbook

[ISO]: ISO/IEC/IEEE 15288:2015 system life cycle processes.

A few 1000s of pages (!) + certification by INCOSE

Why should we (Software Engineers) care?

Systems Engineering (SE) is now a lingua franca for

- Engineering "big" systems, not surprising SE emerged from NASA, DoD, etc
- Managing "big" programs, as acquisition topics are not far
- Can no longer remain the elephant in the room we (ACPS) do not see

Systems engineering

- Relies on engineering concepts and processes: systems science, systems thinking, and ultimately MBSE.
- Vocabulary: models, systems, interfaces, logical, physical, architecture, analysis..

  $\Rightarrow$ To tackle complex challenges

Why should System Engineers care (about software)?

Airbus Gives Alert
(Update3)

By Ed Johnson

Oct. 15 (Bloomberg) -- A
after Australian investig
Ltd. flight switched off t
jet to nosedive.

The Airbus A330-300 wa
computer fed incorrect i
Australian Transport S
650 feet within seconds, slamming passengers and crew into the cabin ceiling, before the pilots regained control.

``This appears to be a unique event,'' the bureau said, adding that Toulouse, France-based Airbus, the world's largest maker of commercial aircraft, issued a telex late yesterday to airlines that fly A330s and A340s fitted with the same air-data computer. The advisory is ``aimed at min

Quantas Airbus A330-300 Forced to make Emergency Landing - 36 Injured

Written by htbw on Oct 7-08 1:40pm ★★★☆☆ Email
From: soyawannaknow.blogspot.com

Thirty-six passengers and crew were injured, some seriously, in a mid-air drama that forced a Qantas jetliner to make an emergency landing, the Australian carrier and police said on Tuesday.

The terrifying incident saw the Airbus A330-300 issue a mayday call when it suddenly changed altitude during a flight from Singapore to Perth, Qantas said.

Two Crashes In Five Months

What's Wrong with Boeing's 737 Max 8?

Boeing's new airplane has only been around for two years and already two 737 Max 8s have crashed, killing 346 people. The disasters may be attributable to a design flaw that emerged when engineers began cutting corners.

Boeing's Max 8 is short, limiting ground clearance under the wings. The engine simply doesn't fit.

FAA says software problem with Boeing 787s could be catastrophic

By **Dan Catchpole**
@dcatchpole

The Federal Aviation Administration says a software problem with Boeing 787 Dreamliners could lead to one of the advanced jetliners losing electrical power in flight, which could lead to loss of control.

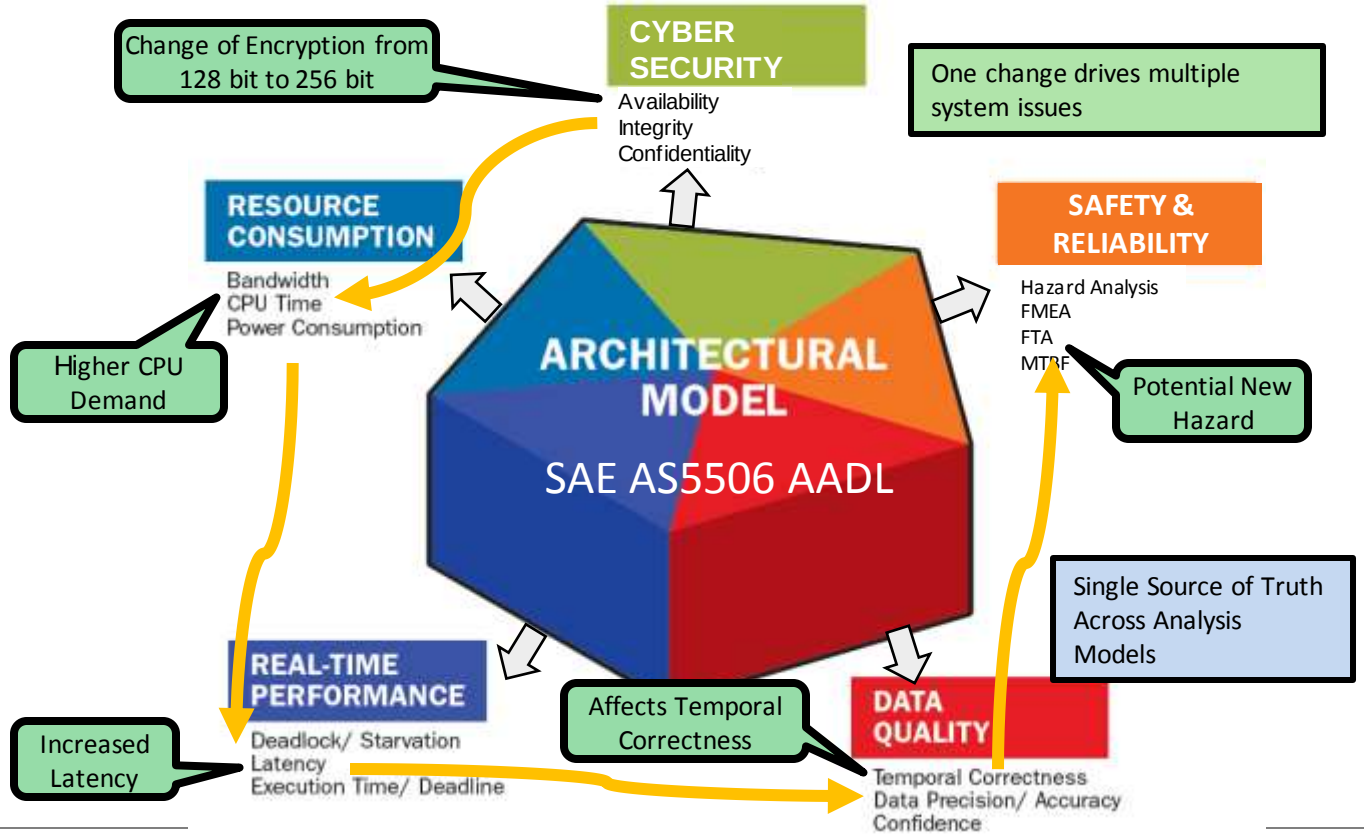
- **The Buzz**: Hipster's dilemma
- Boeing & aerospace news
- Aerospace blog

The FAA notified operators of the airplane Friday that if a 787 is powered continuously for 248 days, the plane will automatically shut down its alternating current (AC) electrical power.

Embedded software systems introduce a new class of problems not addressed by traditional system safety analysis

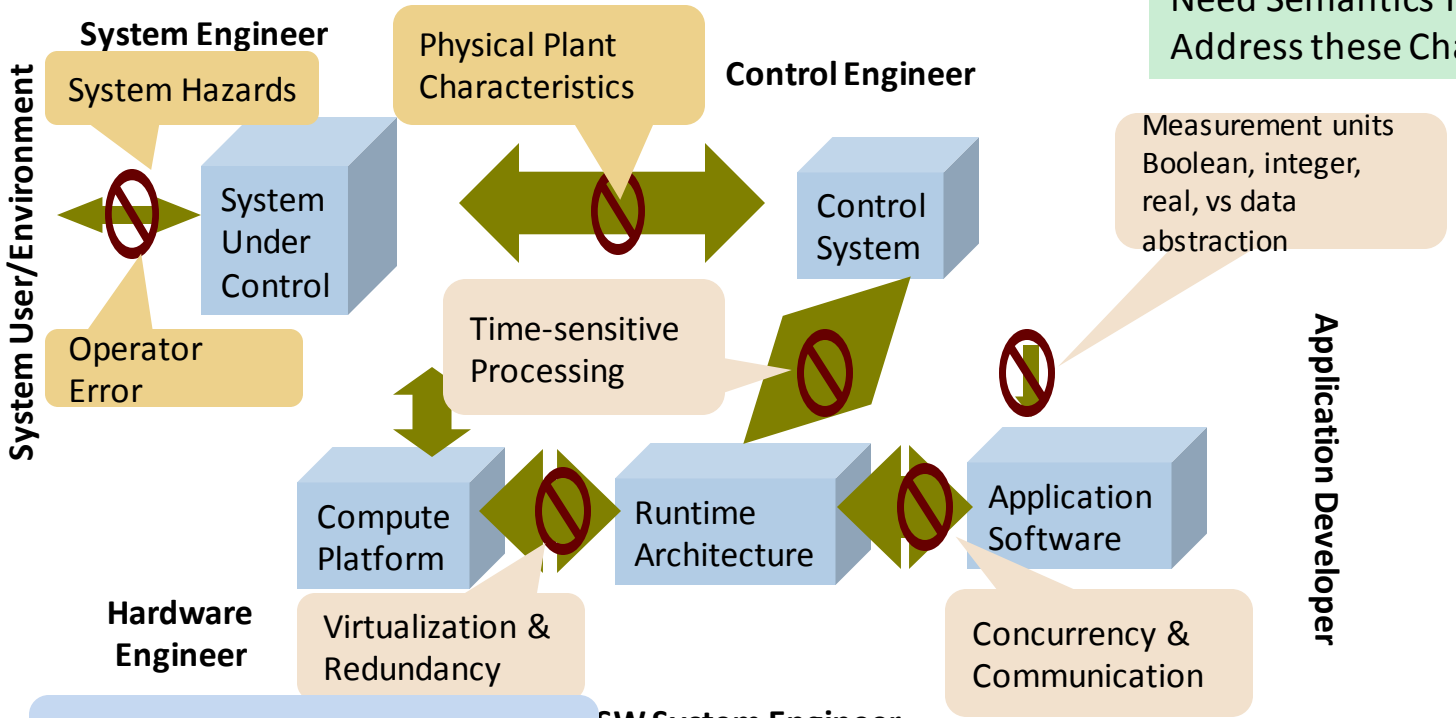
Breakdown in human intensive safety assessment process

Architecture as integration focal point – Software systems



Architecture as integration focal point – Cyber-Physical systems

Need Semantics To Address these Challenges



Why do system level failures still occur despite best safety practices?

*Cyber-Physical Systems have become a major **safety** and **cyber security** risk*

On processes – ISO15288

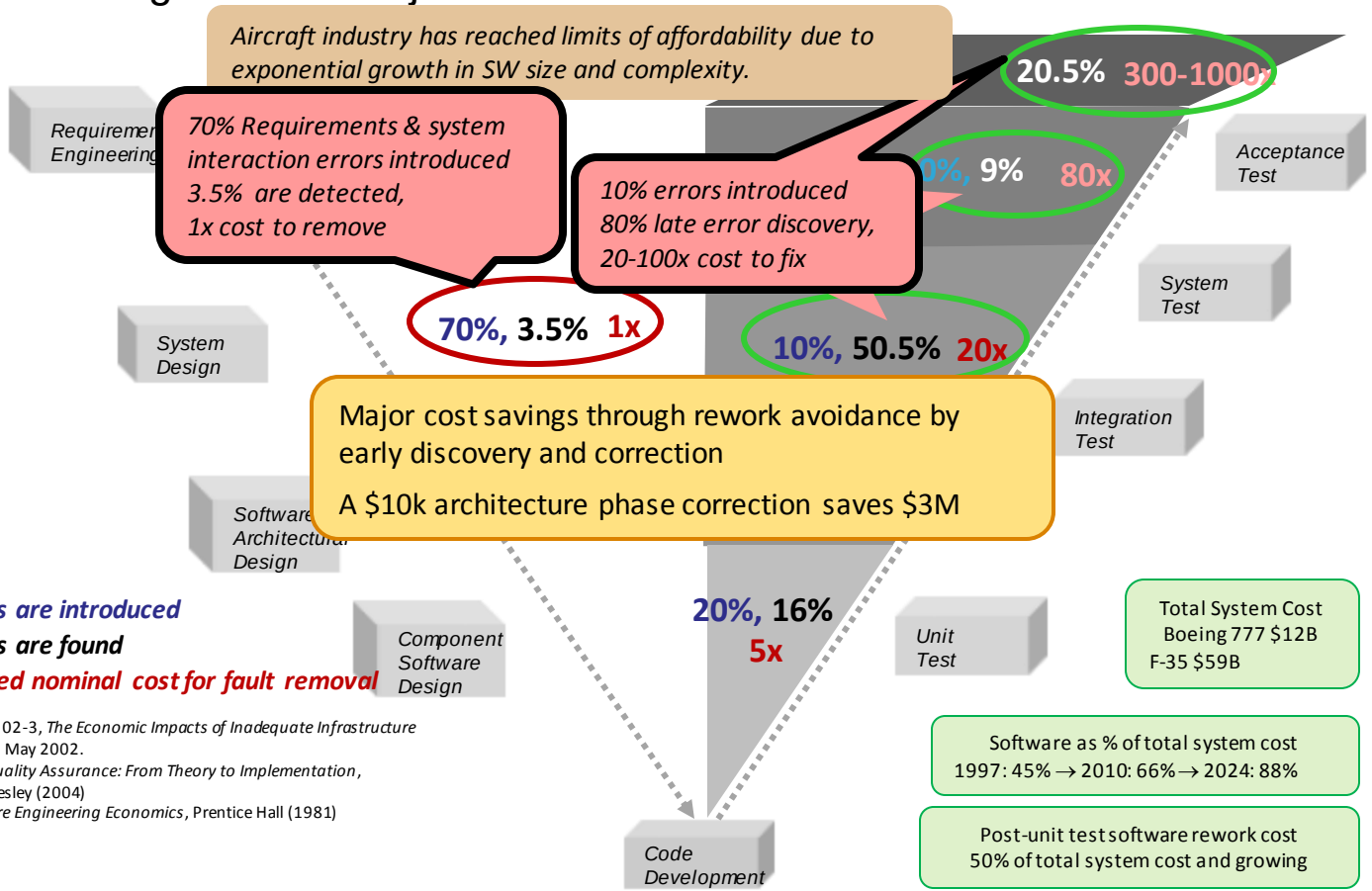
A process is a set of interrelated or interacting activities which transforms inputs into outputs (ISO 9000:2005)

Process are organized so that they contribute to a specific goal. They are structured by an *argumentation* that justifies the existence of each step.

Technical processes may leverage architectural models !

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**.

High Fault Leakage Drives Major Increase in Rework Cost



Sources:
NIST Planning report 02-3, *The Economic Impacts of Inadequate Infrastructure for Software Testing*, May 2002.
D. Galin, *Software Quality Assurance: From Theory to Implementation*, Pearson/Addison-Wesley (2004)
B.W. Boehm, *Software Engineering Economics*, Prentice Hall (1981)

The Safety-Critical Embedded Software System Challenge

Problem:

Software increasingly dominates safety and mission critical system development cost.

80% of issues discovered post unit test.

Solution: Early discovery of system level issues through architecture modeling, virtual Integration and incremental analytical assurance.

Approach:

International standard based research driven technology matured into practice through pilot projects and industry initiatives. (SAE International Aerospace Standard AS-5506B)

Development of an open source research prototyping platform continually enhances analysis, verification, and generation capabilities.

Reducing Defect Leakage through Early Analytical Assurance is Critical

Outline

Model Based Engineering at the SEI

Model-Based Systems Engineering and the impact of Architecture

Model-Based System Engineering, standards, and AADL

Model-Based System/Software Engineering Overarching objectives

MBSE complements typical software programming with **models** to

1. Organize stakeholders needs and elicit requirements
2. Capture system elements – design, reverse engineering or COTS
   - Interfaces, components internals (static and behavioral), and
   - a system architecture built from those: deployment, (re-)configuration
3. Apply analytical frameworks to assess model's "compliance to some objectives"
   - Syntactic, conformance to guidelines, patterns
   - Quality of system, w.r.t. performance, safety, security, behavior metrics
4. Synthesize portions of software from models
   - E.g. functional: Simulink, SCADE; Architectural: UML, AADL
   - No synthesis or link to code in SysML, as SysML has only high-level concepts

Models as processable artifacts to guide the software engineering process

"Modeling is the new programming"

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

MBSE – Not just SysML

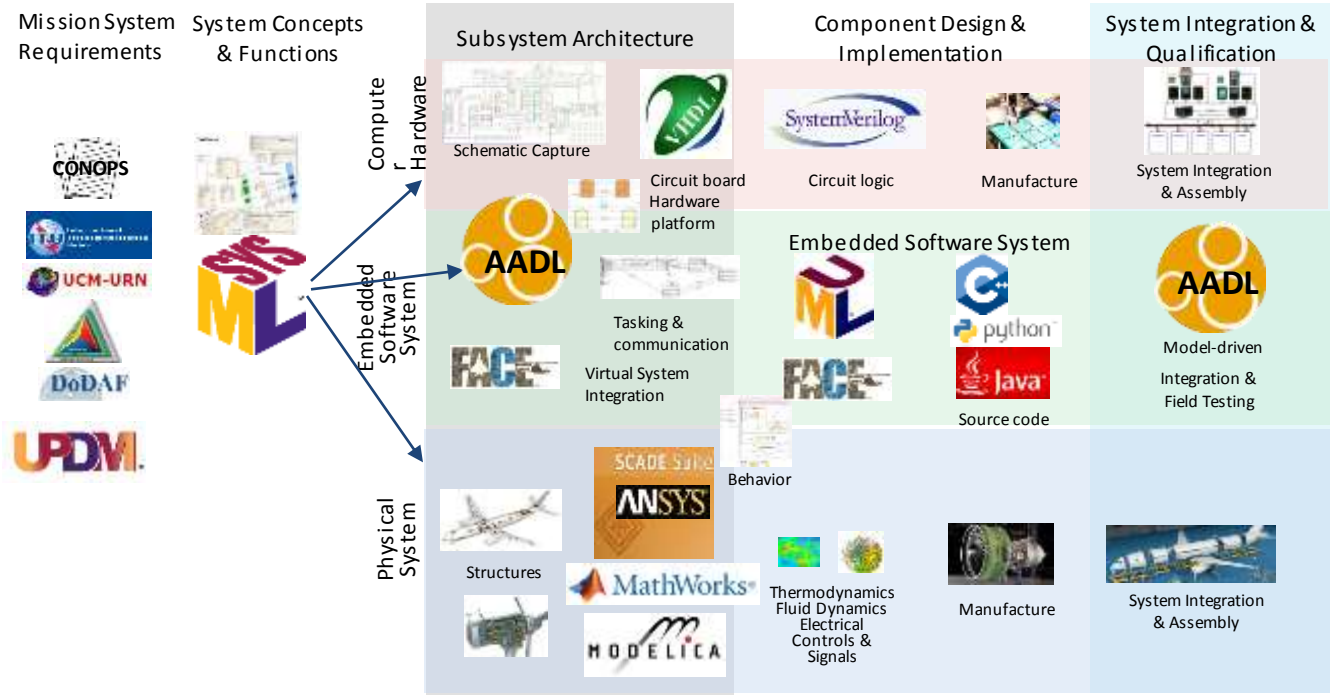
Model-based systems engineering (MBSE) is the *"formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases."* (INCOSE 2007)

SysML support capturing relationships among system functions, requirements, developers, and users. But not the later development stages

Other modeling notations are required to

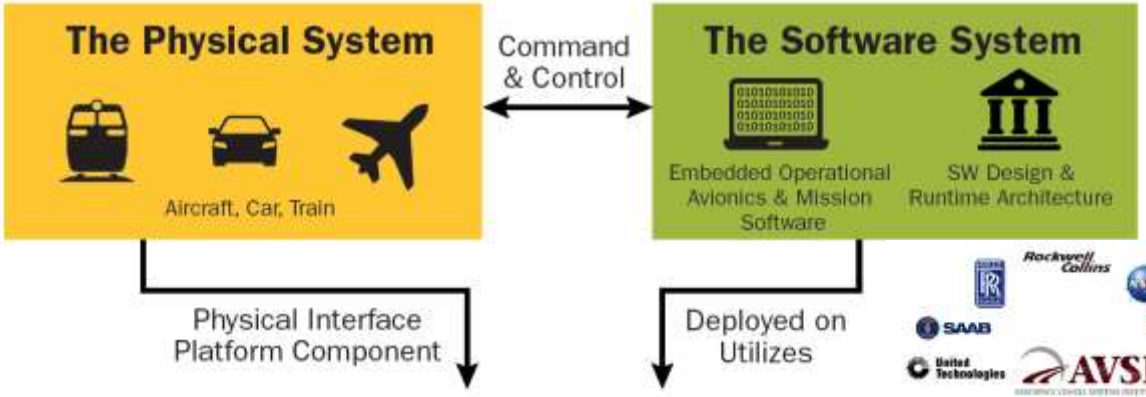
> Capture hardware platforms, software architecture, behavioral semantics, deployment of SW to HW, support safety or security assessment, performance analysis, behavioral verification, memory budget validation, etc...

Not just SysML vs AADL, larger aggregation of standards



Filling the Modeling and Analysis Gap for Embedded Software System

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



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

Outline

Model Based Engineering at the SEI

Model-Based Systems Engineering and the impact of Architecture

Model-Based System Engineering, standards, and AADL

AADL Language Overview

AADL Standard Suite (AS-5506 series)

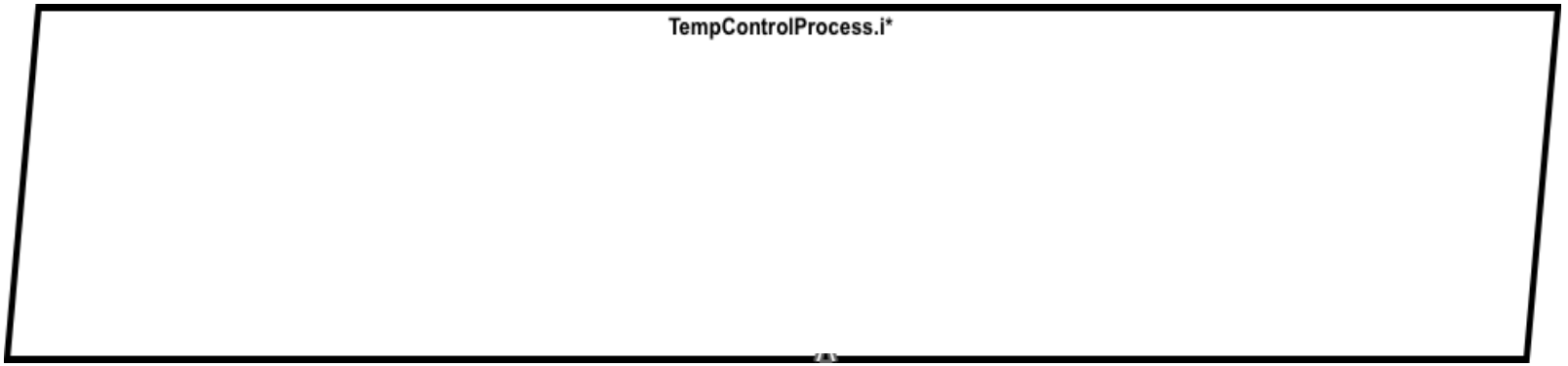
Core AADL language standard vV1 2004, .. v2.3 2022]

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

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]

AADL Overview

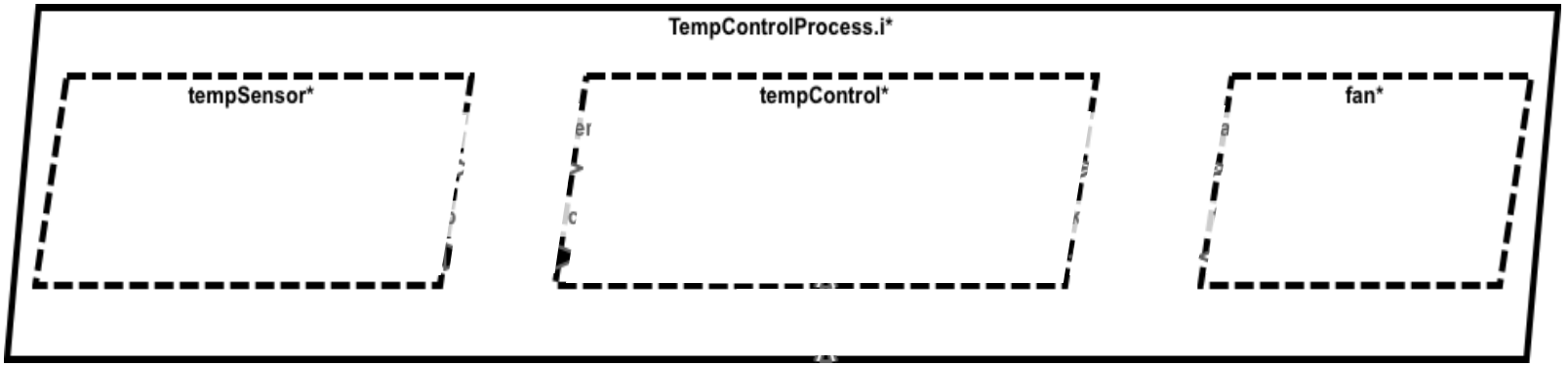


Like a lot of models that engineers draw every day on their whiteboards, AADL consists of boxes and lines

The difference between AADL and a whiteboard is that AADL has precise *semantics*

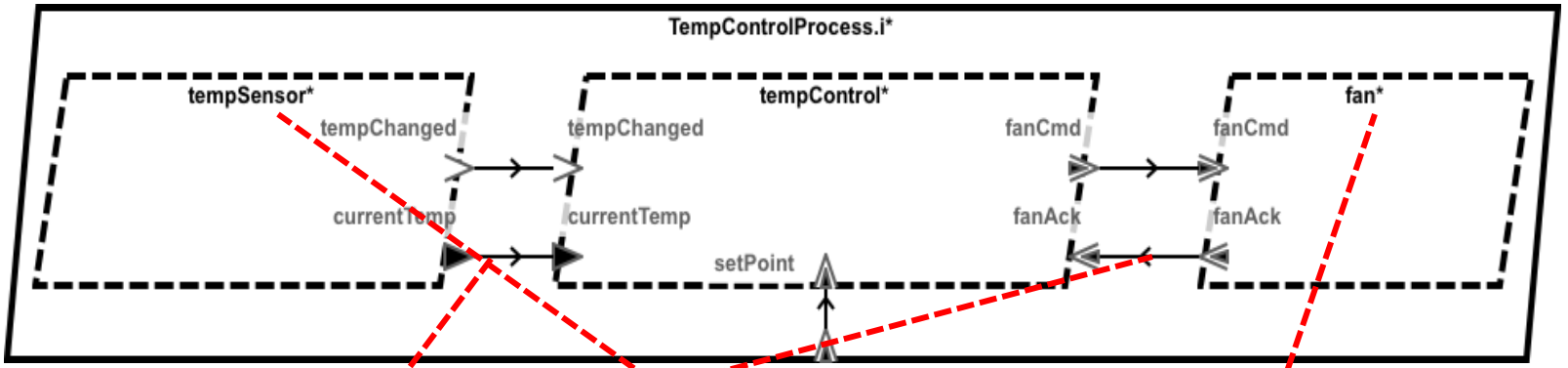
This box represents a computer process – a protected region of memory and a space where we can allocate individual threads

AADL Overview



Those threads are also boxes – but they have very precise meanings.

AADL Overview



We can connect the threads together using lines to represent different types of intra-process communication

This box shows a periodic thread – it is dispatched regularly according to some clock

We add more semantics via *properties* – they are useful for both system analyses and to guide code generation

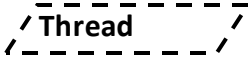
And this thread is sporadic – it is dispatched whenever a message arrives at a specified port

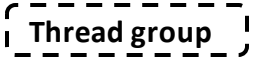
What are AADL Components?

Application Components

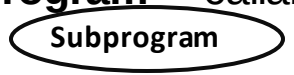
System – hierarchical organization of components 

Process – protected address space 

Thread – a schedulable unit of concurrent execution 

Thread group – logical organization of threads 

Data – potentially sharable data 

Subprogram – callable unit of sequential code 

Execution Platform & Device Components

Processor / Virtual Processor – Provides thread scheduling and execution services 

Memory – provides storage for data and source code 

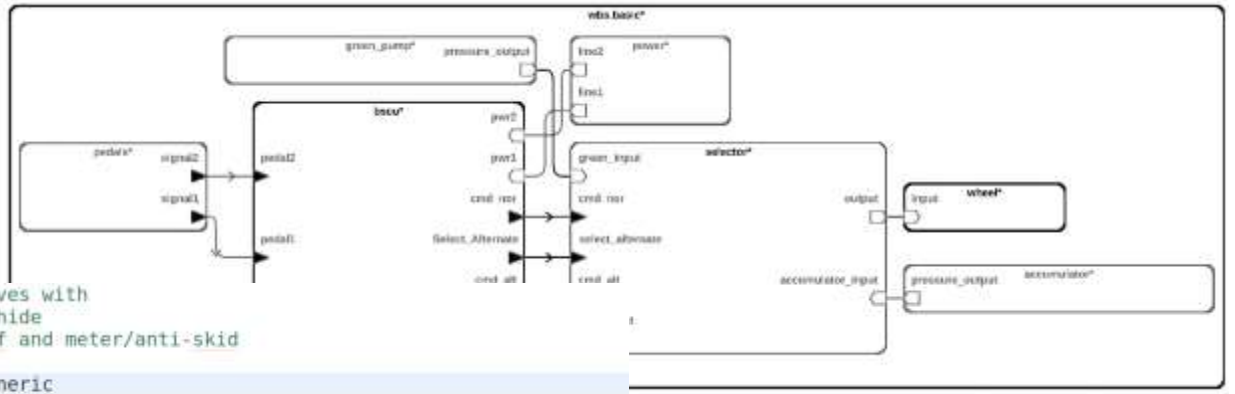
Bus / Virtual Bus – provides physical/logical connectivity between execution platform components 

Device – interface to external environment 

What does AADL actually look like?

Semi-formal semantics

Only architectural elements

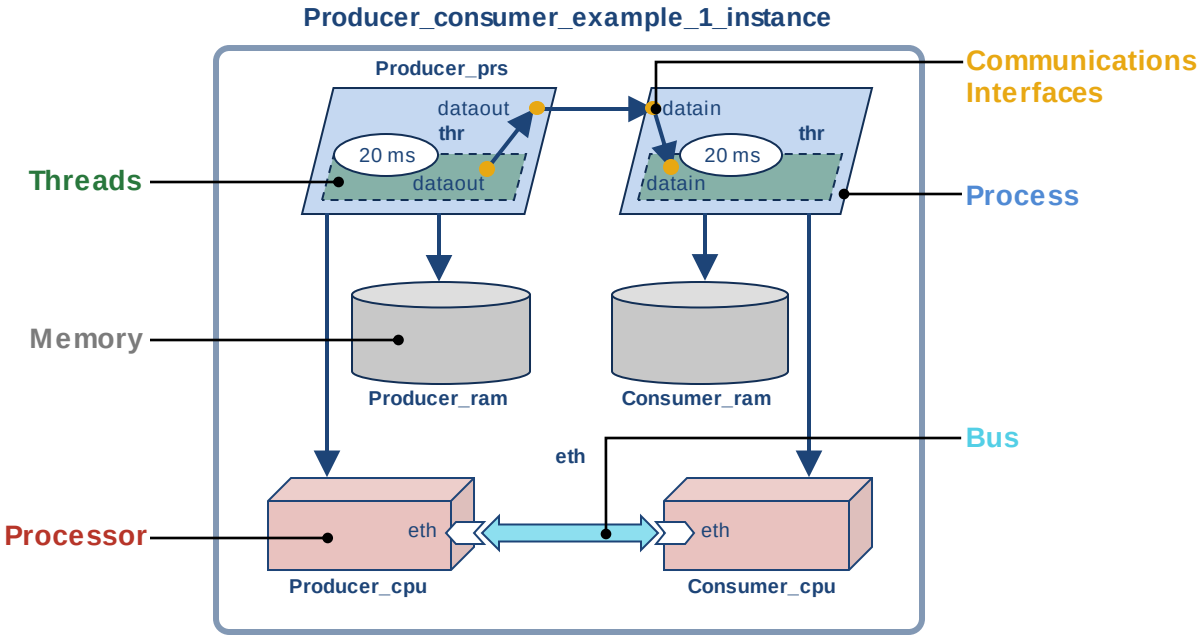


```
79 --  Basic/naive version that abstracts all the valves with
80 --  a selector_subsystem. This selector subsystem hide
81 --  the physical logic behind the selector, shutoff and meter/anti-skid
82 --  valves.
83     system implementation wbs.basic extends wbs.generic
84 subcomponents
85     bscu : refined to system impl::bscu::bscu.basic;
86     --  The selector subsystem
87     selector   : refined to system impl::valves::selector_basic{Classifier_Substitution_Rule => Type_Exte
88     wheel      : refined to system impl::wheel::wheel_one_input.i{Classifier_Substitution_Rule => Type_Ex
89 connections
90     blue_to_selector    : bus access blue_pump.pressure_output  <-> selector.blue_input;
91     green_to_selector   : bus access green_pump.pressure_output <-> selector.green_input;
92
93     bscu_sel_to_selector : port bscu.Select_Alternate -> selector.select_alternate;
94     bscu_cmdnor_to_selector : port bscu.cmd_nor -> selector.cmd_nor;
95     bscu_cmdalt_to_selector : port bscu.cmd_alt -> selector.cmd_alt;
96
97     selector_to_wheel   : bus access selector.output <-> wheel.input;
98 end wbs.basic;
```

Annexes add functionality:

- Error Modeling
- Behavior
- Code Generation

An Example Model (graphical)



This AADL model represents threads executing within processes (dedicated address spaces), the communications and connections among the components, and the binding of threads and processes to computer resources (e.g., threads to the processor on which they execute).

Outline

Model Based Engineering at the SEI

Model-Based Systems Engineering and the impact of Architecture

Model-Based System Engineering, standards, and AADL

AADL Language Overview

AADL Tooling

AADL capabilities

AADL is highly tunable, with a restricted set of concepts
- Demonstrated many use cases, 1600+ academic publications

AADL as a backbone, federating multiple activities
- analysis through generation of intermediate models + external tools

Non exhaustive list of analysis capabilities

- Integration: SysML, FACE, Simulink, SCADE
- Architectural pattern checks:
  MILS, ARINC, Ravenscar, Synchronous
- Model checking:
  - Timed/Stochastic/Colored Petri Nets
  - Timed automata et al.: UPPAAL, Versa, TASM
- Scheduling: OSATE, CAMET, MAST, Cheddar, CARTS
- Performance evaluation: real-time and network calculus
- Fault analysis: OSATE, COMPASS,
  Mapping to Stochastic Petri Nets, PRISM
- Security: CAMET (DoDI 8510.01)
- Simulation: ADeS, Marzhin
- Energy consumption of SoC: OpenPeople project
- Code generation: SystemC, C, Ada, RTSJ, Lustre
- WCET analysis: mapping to Bound-T

AADL demonstrated its suitability to support various analysis for the real world

AADL commercial and open source toolchains

Multiple AADL toolchains exist, they can be easily combined thanks to the textual syntax.

- **OSATE** (SEI/CMU) https://osate.org/
  - Eclipse-based tools. Reference implementation
  - Textual and graphical editors + various plug-ins for latency, processor utilization, memory utilization, data consistency, security, safety analysis (MIL STD 882E, ARP4761), ARINC653
- **CAMET** (Adventium Lab) https://www.adventiumlabs.com/curated-access-model-based-engineering-tools-camet-library
  - Extensions to OSATE to support other analysis (Multiple Independent Levels of Security (MILS), Framework for Analysis of Schedulability, Timing and Resources (FASTAR)
- **SCADE Architect** (ANSYS Esterel) https://www.ansys.com/products/embedded-software/ansys-scade-suite
  - Eclipse-based tools. Combine SysML, AADL and other formalisms, code generation
- **AADL Inspector** (Ellidiss) https://www.ellidiss.com/products/aadl-inspector/
  - Lightweight editor, model simulation, scheduling analysis

Outline

Model Based Engineering at the SEI

Model-Based Systems Engineering and the impact of Architecture

Model-Based System Engineering, standards, and AADL

AADL Language Overview

AADL Tooling

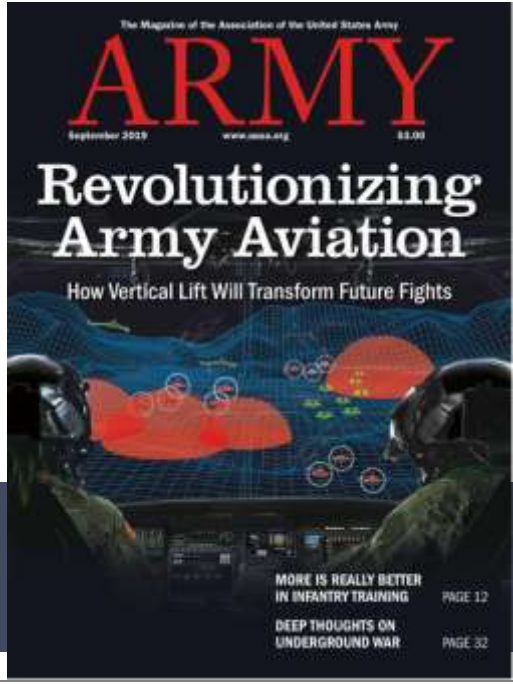
AADL transition stories

Helping to Revolutionize Army Aviation

Over many years, the SEI has had an outstanding partnership with the US Army, who is at the vanguard of applying AADL and ACVIP to the Army's future vertical lift challenge.

Benefits of AADL & ACVIP (via Alex Boydston)

- Decreased fielding time by finding problems early
- Early risk reduction by discovering performance issues early
- Increased cybersecurity by using AADL/ACVIP to improve system security
- Decreased development costs and support for MOSA and certification by transforming procurement supporting MBE and ACVIP



Virtual integration of software, hardware, and system supports verification, airworthiness, safety, and cybersecurity certification

Impact

Finding Problems Early (AMRDEC/SEI)

Summary: 6 week virtual integration of health monitoring system on CH47 using AADL

Result: Identified 20 major integration issues early

Benefit: Avoided 12-month delay on 24 month program

CH47 Chinook

DARPA

High Assurance Cyber Military Systems (HACMS)

Unmanned Quadcopter

Unmanned Little Bird

TARDEC Autonomous Truck

Improving System Security (DARPA/AFRL)

Summary: AADL applied to unmanned aerial vehicles & autonomous truck

Result: AADL models enforced security policies and were used to auto-build the system

Benefit: Combined with formal methods verification, prevented security intrusion by a red team

---

Transforming procurement (Joint Multi-Role)

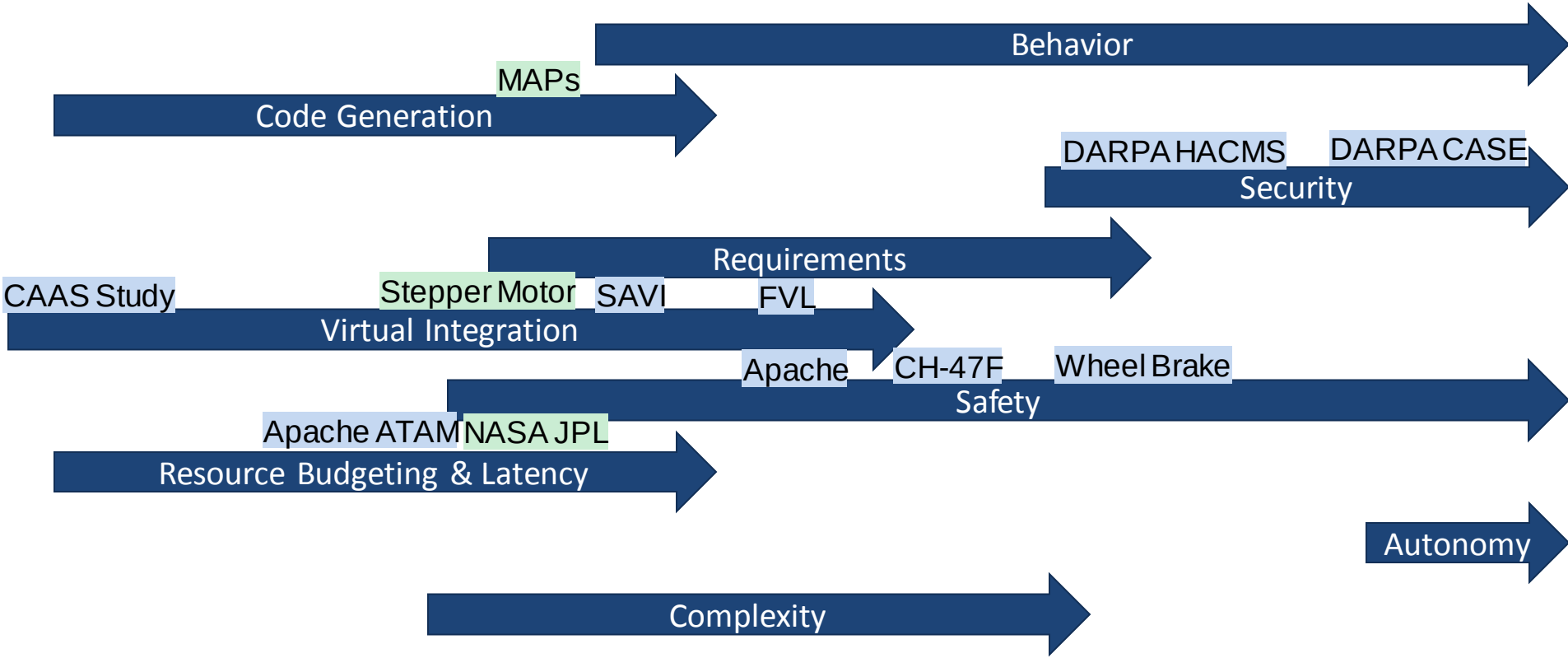
Summary: Industry/DoD process demonstration

Result: Pre-integration fault identification

Benefit: 10X reduction integration test cost

Makes complex capabilities possible through Agile analytic and virtual integration of real-time safety and security critical cyber physical embedded systems

A Holistic View of Lines of Research Enabled by AADL



Outline

Model Based Engineering at the SEI

Model-Based Systems Engineering and the impact of Architecture

Model-Based System Engineering, standards, and AADL

AADL Language Overview

AADL Tooling

AADL transition stories

Research topics: ModDevOps, ETMAC

From DevOps to ModDevOps

DevOps delivers software faster with increased quality:

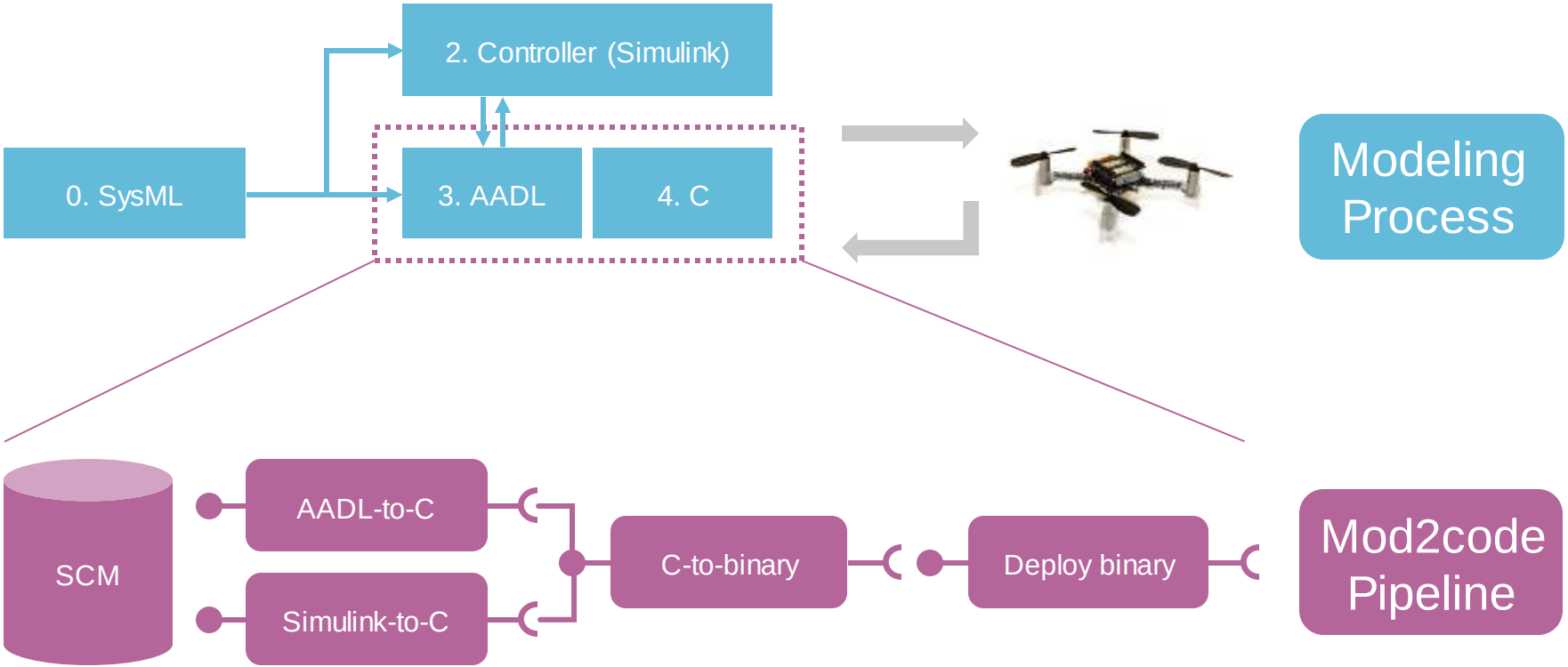
- Continuous integration/deployment
- Containerized systems

DevOps is a software process, to be adapted to systems.

"ModDevOps is a systems/software co-engineering culture and practice that aims at unifying systems engineering (Mod), software development (Dev) and software operation (Ops). The main characteristic of the ModDevOps is to strongly advocate abstraction, automation, and monitoring at all steps of system construction."

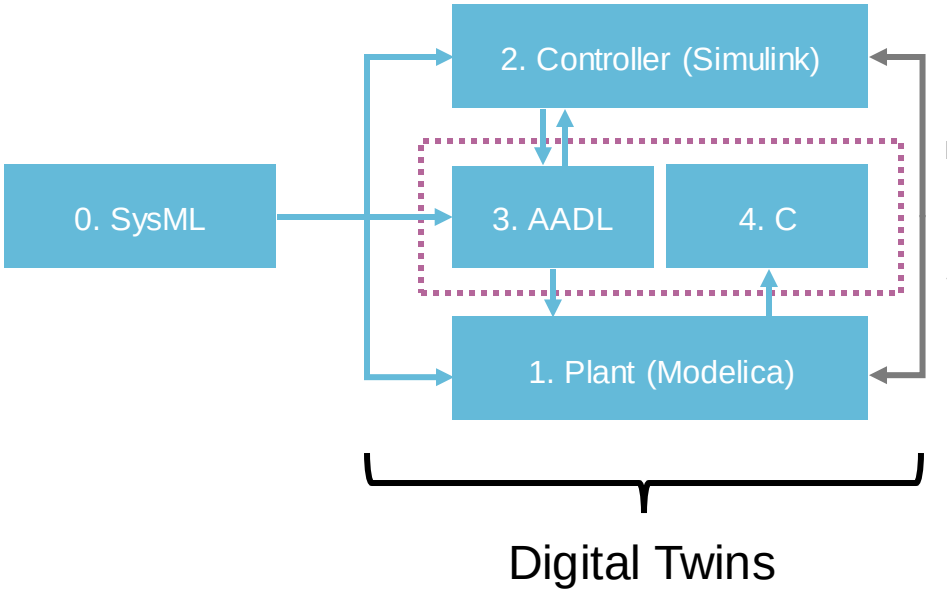
(adapted from https://software.af.mil/training/devops/)

ModDevOps in Action – Modeling Process



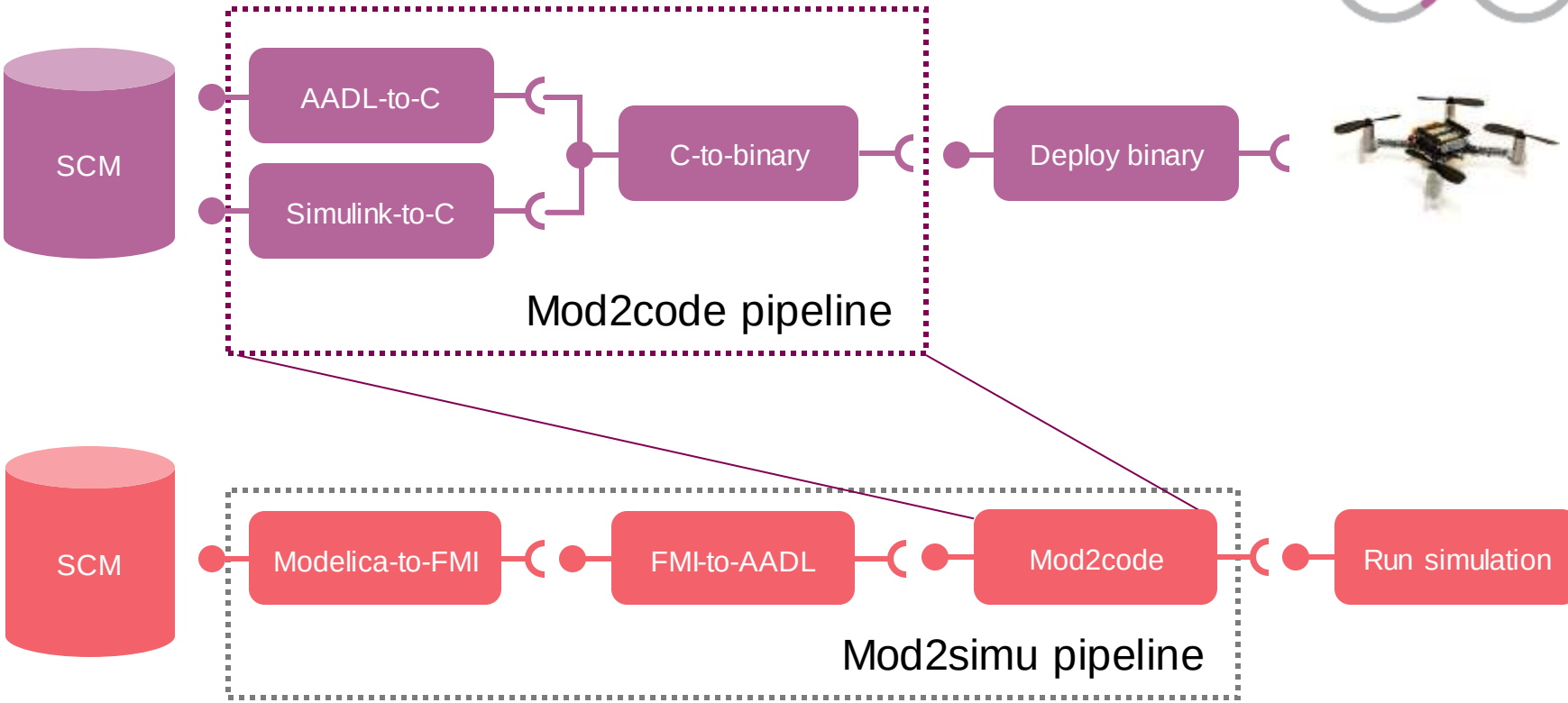
From ModDevOps to TwinOps

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.

ModDevOps in Action – ModDevOps Pipeline #2



Ensuring Transition of MBE with Assurance Contracts

Digital Engineering: Model-Analyze-Build

Late Discovery of Design Errors in DoD Systems is very costly.

Architecture modeling and analysis can **detect** design **error early**

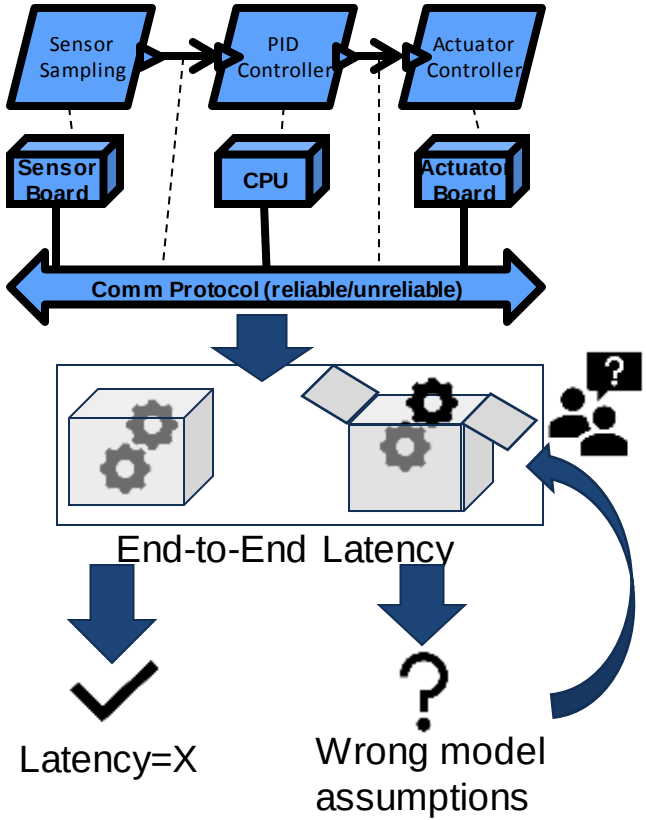
BUT:

Analysis assumptions are often implicit

if analysis **assumptions not met**: analyses break down for reasons not clear to users of analysis tools.

E.g., e2e Latency Assumption: periods multiple of each other (harmonic)

DoD barrier for adoption

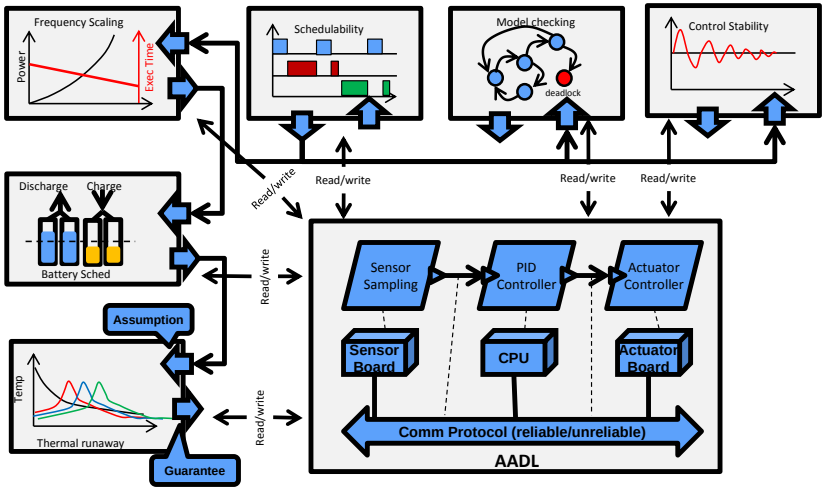


Digital Engineering: Multiple Claims - Multiple Analyses

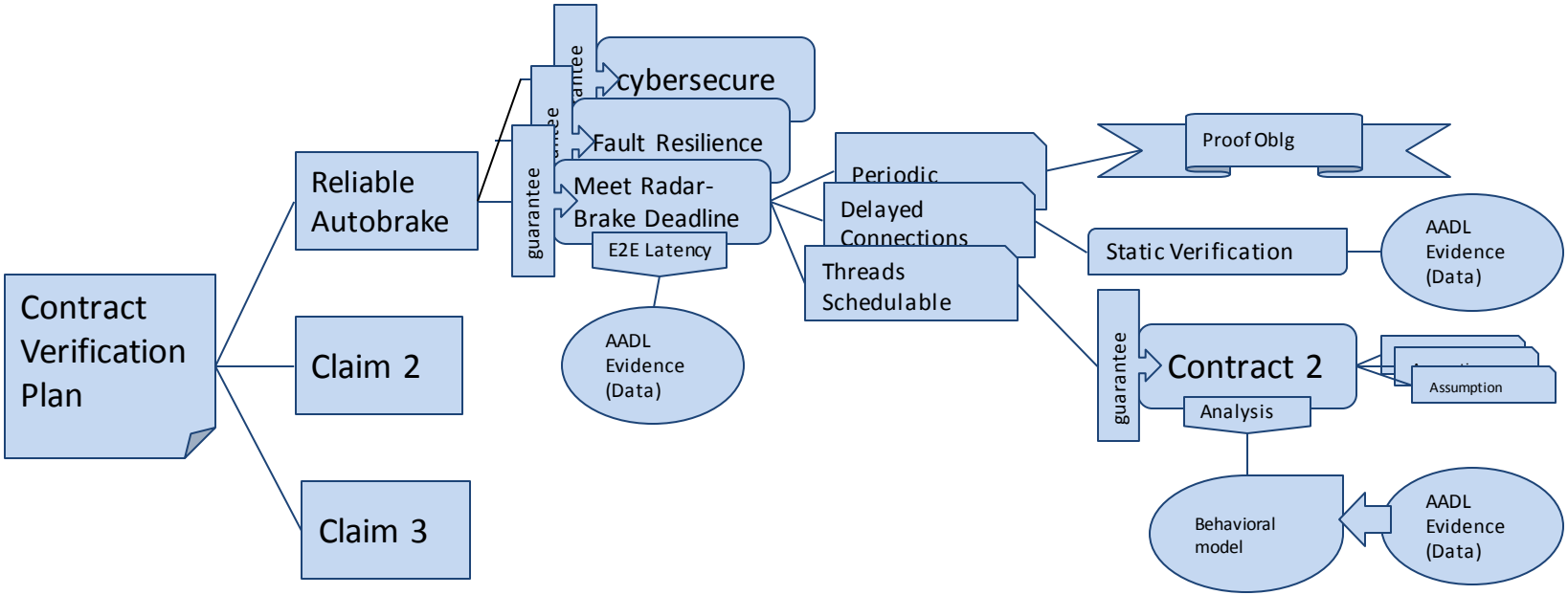
Different Assurance Claims

- Combine multiple analysis
- Validate assumptions
- Resolve assumption conflicts

Integrate into arguments to satisfy claims



Assurance Contract Argumentation



Shift Left And Down to the Metal

Early Analysis

- Evaluate design decisions with partial information
- E.g., latency analysis before worst-case execution time (WCET)
  - periods of tasks must be multiples of each other

Refinement

- Track pending information
  - WCET
- Track and execute pending verification
  - Schedulability

Conformance

- Track implementation assumption
- Verify implementation conformance
  - Task executed strictly periodic $\alpha_{L,3}$

Outline

Model Based Engineering at the SEI

Model-Based Systems Engineering and the impact of Architecture

Model-Based System Engineering, standards, and AADL

AADL Language Overview

AADL Tooling

AADL transition stories

Conclusion

Wrap-Up

Model-based systems engineering (MBSE) is paramount to build DoD complex systems.

There is no silver bullet, projects need multiple languages

Capturing relationships among system functions, requirements, stakeholders down to

Prediction of runtime characteristics, e.g. performance, safety

⇒ SysML and AADL and …

AADL supports safety or security assessment, performance analysis, behavioral verification, memory budget validation, etc.

Many videos and technical reports available in the SEI Digital Library