ADEPT Workshop 2024
AADL Intro and News

Bruce Lewis, AADL Committee Chair

Galois, Inc.

June 14, 2024

galois

Architecture Analysis & Design Language (AADL) History & Objectives

- Came out of 3 DARPA programs (9 years plus) developing Architecture Design Languages as MetaH (Steve Vestal PI)
- Experiments in Army Lab prove value so started SAE standard for AADL from MetaH
- System Architecture Virtual Integration (SAVI), an industry program, selected AADL after review of all competitors to deal with the high cost of system of aviation software integration.
- DARPA programs have since leveraged AADL (HACMS, CASE, PROVERS …) combined with formal methods
- AADL Language Architect – Peter Feiler -> Jerome Hugues
- Key concepts of AADL from the beginning – Domain specific Language for RT embedded systems
  - Enable quantitative architectural analysis on virtually integrated systems.
  - Enable generative approaches to build compliant systems from verified models.
  - Provide stable core concepts and language with well defined semantics
  - Easy to understand engineering terms with textual and graphical expression
  - Incremental refinement to support the lifecycle with incremental analyzability
  - Flexibility to support new domains & analyses w annex sublanguages, property sets.

Architecture Analysis & Design Language
SAE International AS-5506 STANDARD SUITE

- **Core AADL language standard upgrades**
- **V1 [A) 2004, V2 (B) 2012, V2.2 (C) 2017, V2.3 (D) 2022**
  - For embedded & cyber physical software system modeling, analysis, and generative integration
  - Strongly typed component based architecture language with well-defined, rich semantics for threads, processes on partitions, subprograms and processor, memory, bus, system and device components, sampled/queued, communication, modes, end-to-end flows
- **Next standard will be joint SAE/OMG standard AADL library for SysMLv2**

- **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 components and interaction [2011, 2017]
  - Data Modeling extension for interfacing with data models (UML, ASN.1, …) [2011]
  - AADL Runtime System & Code Generation [2006, 2015, RTS refined in Core in 2022]

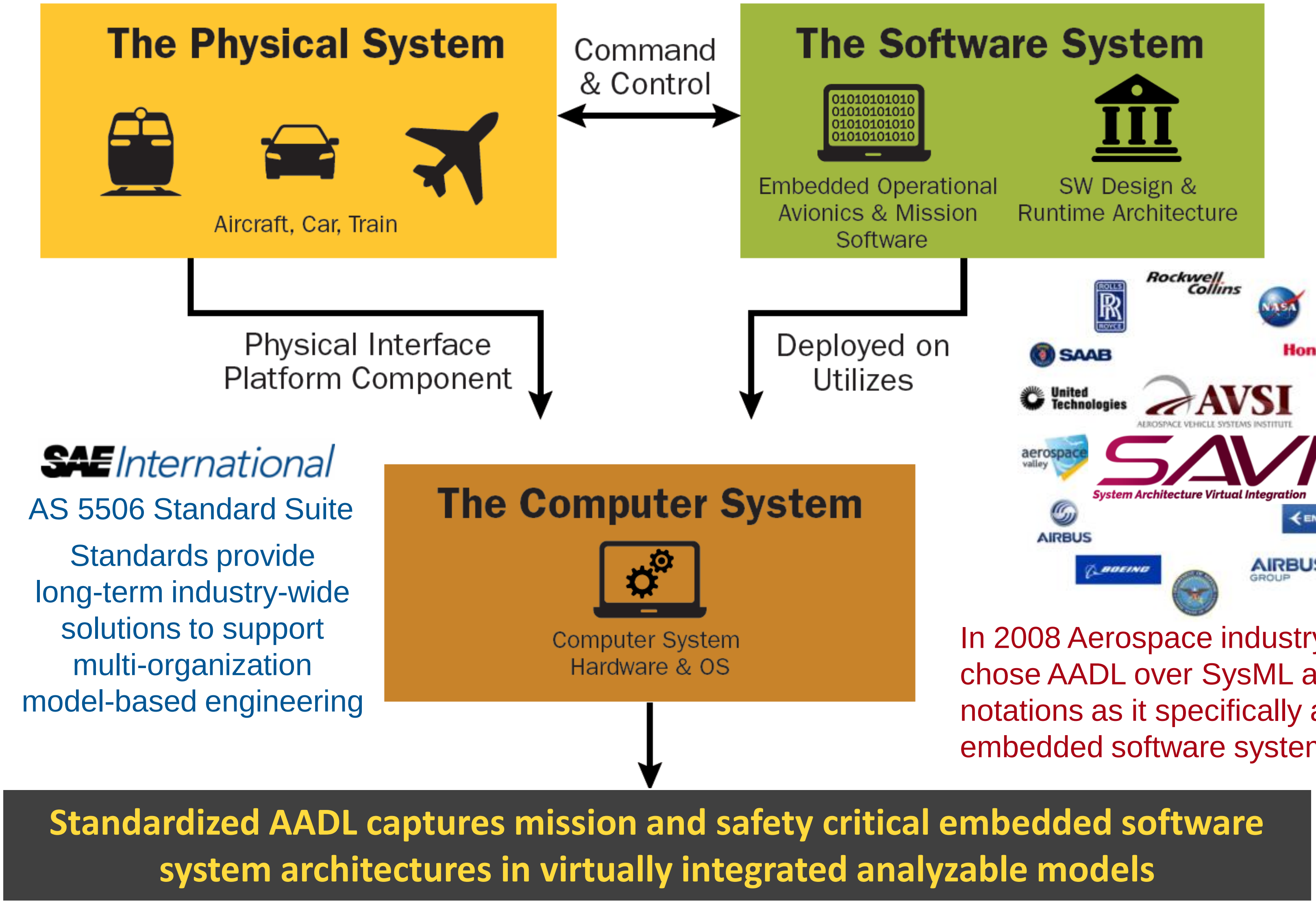
AADL SAE AS2C Committee Activities

- Current focus is developing a SysMLv2 library for AADL
  - It is planned to be a joint OMG/SAE standard
  - It will make AADL part of SysMLv2 as a supported library integrating system engineering and embedded system design.
  - Part of OMG's Systems Modeling Community (SMC)
  - Our SMC is "Real-Time Embedded Safety-Critical Systems Working Group (RTESCWG)
  - SAE and OMG working together to formalize the joint standardization process. Both parties working together well.
  - Involves coordinated upgrades to SysMLv2 to support real time systems
  - Jerome Hughes and Gene Shreve are co-chairs of the Real Time SMC

- OMG/SAE Joint meetings
  - Two virtual meetings per month
  - Typically every other Wed 9:00-10:00 CT, Next meeting June 19th
  - You can join the OMG Managed Communities or the SAE AADL committee to attend virtual meetings.
  - To attend SMC meetings at OMG standards meetings, you need to join the SMC.
  - Next OMG standards meeting, Chicago, USA, Sept 11-12

- Progress – Static part of AADL prototyped, being used on PROVERS

AADL ANALYTICALLY DESCRIBES THE REAL-TIME SYSTEM ENABLING VIRTUAL INTEGRATION


The Physical System
Aircraft, Car, Train
Command & Control
The Software System
Embedded Operational Avionics & Mission Software
SW Design & Runtime Architecture
Physical Interface Platform Component
Deployed on Utilizes
The Computer System
Computer System Hardware & OS

SAE International

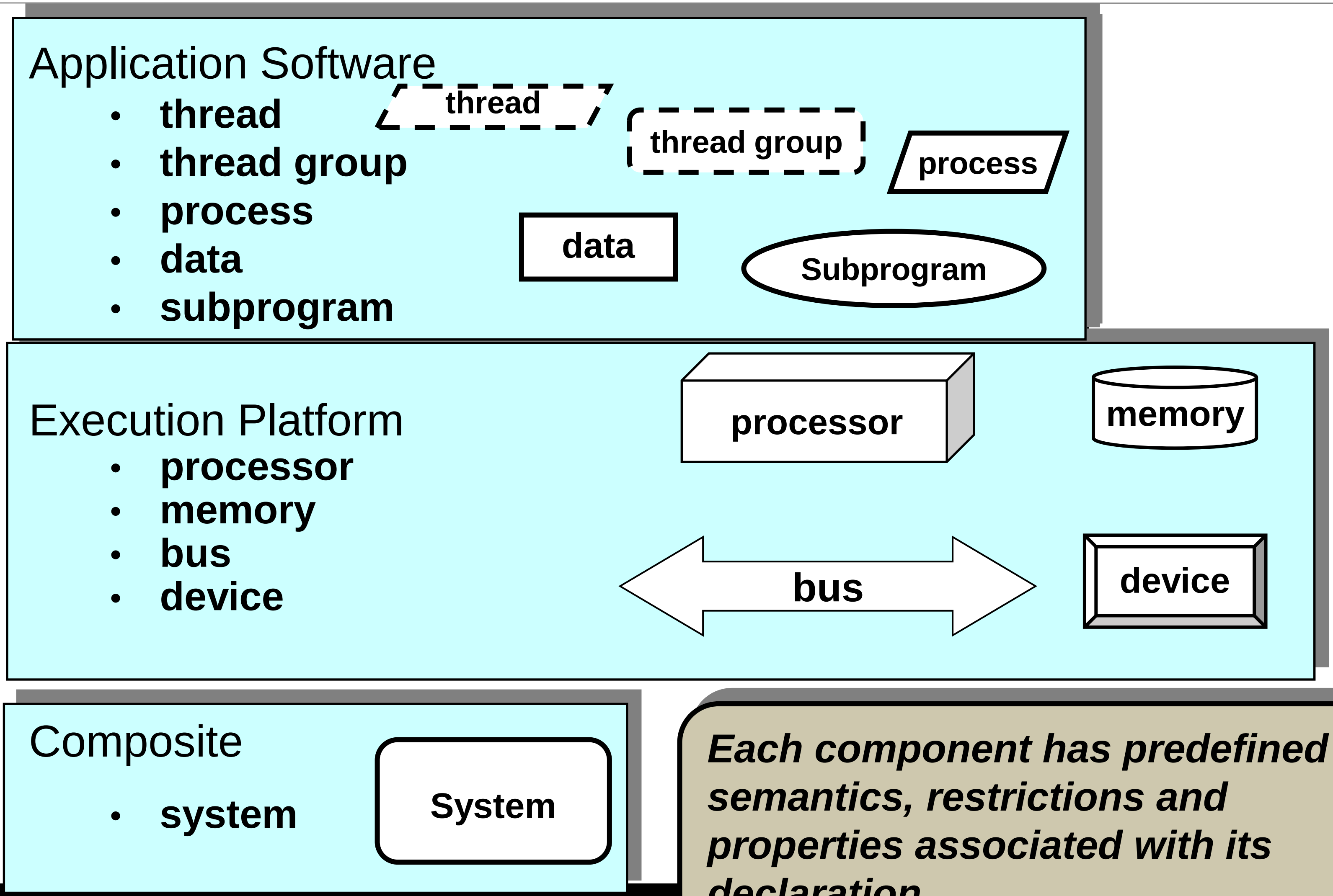
AS 5506 Standard Suite

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

Standardized AADL captures mission and safety critical embedded software system architectures in virtually integrated analyzable models

AADL Components



Application Software

- thread
- thread group
- process
- data
- subprogram

Execution Platform

- processor
- memory
- bus
- device

Composite

- system

Each component has predefined semantics, restrictions and properties associated with its declaration.

Ports & Connections

Ports: directional transfer of data & control

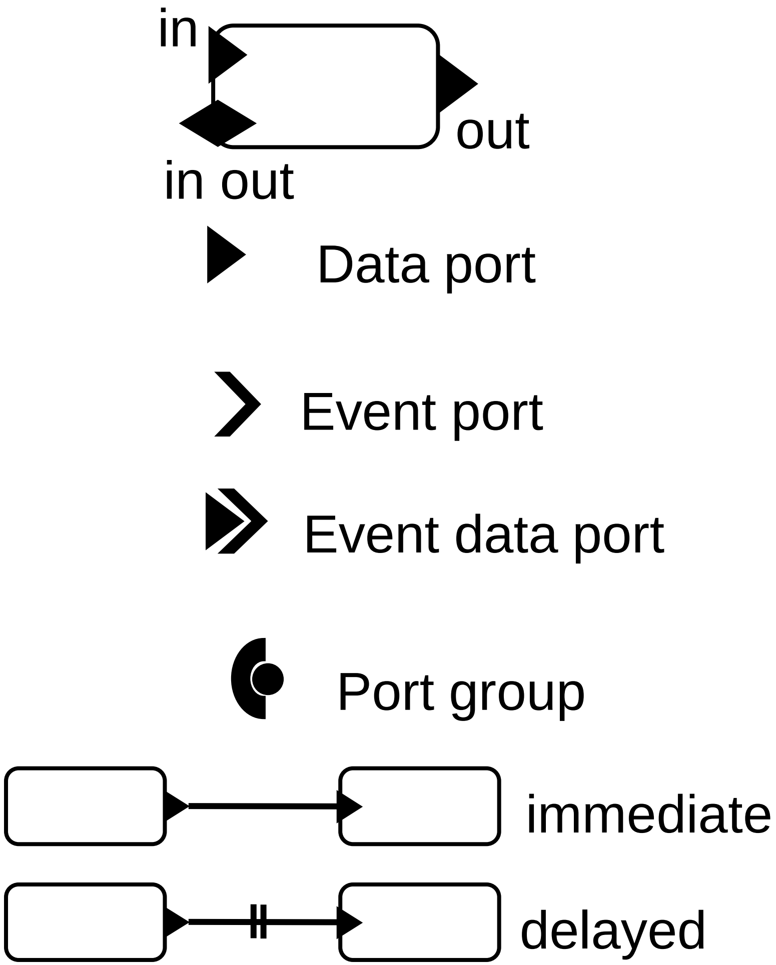
Data port: state, sampled data streams

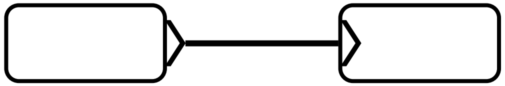
Event port: Queued, thread dispatch & mode switch trigger

Event data port: queued messages

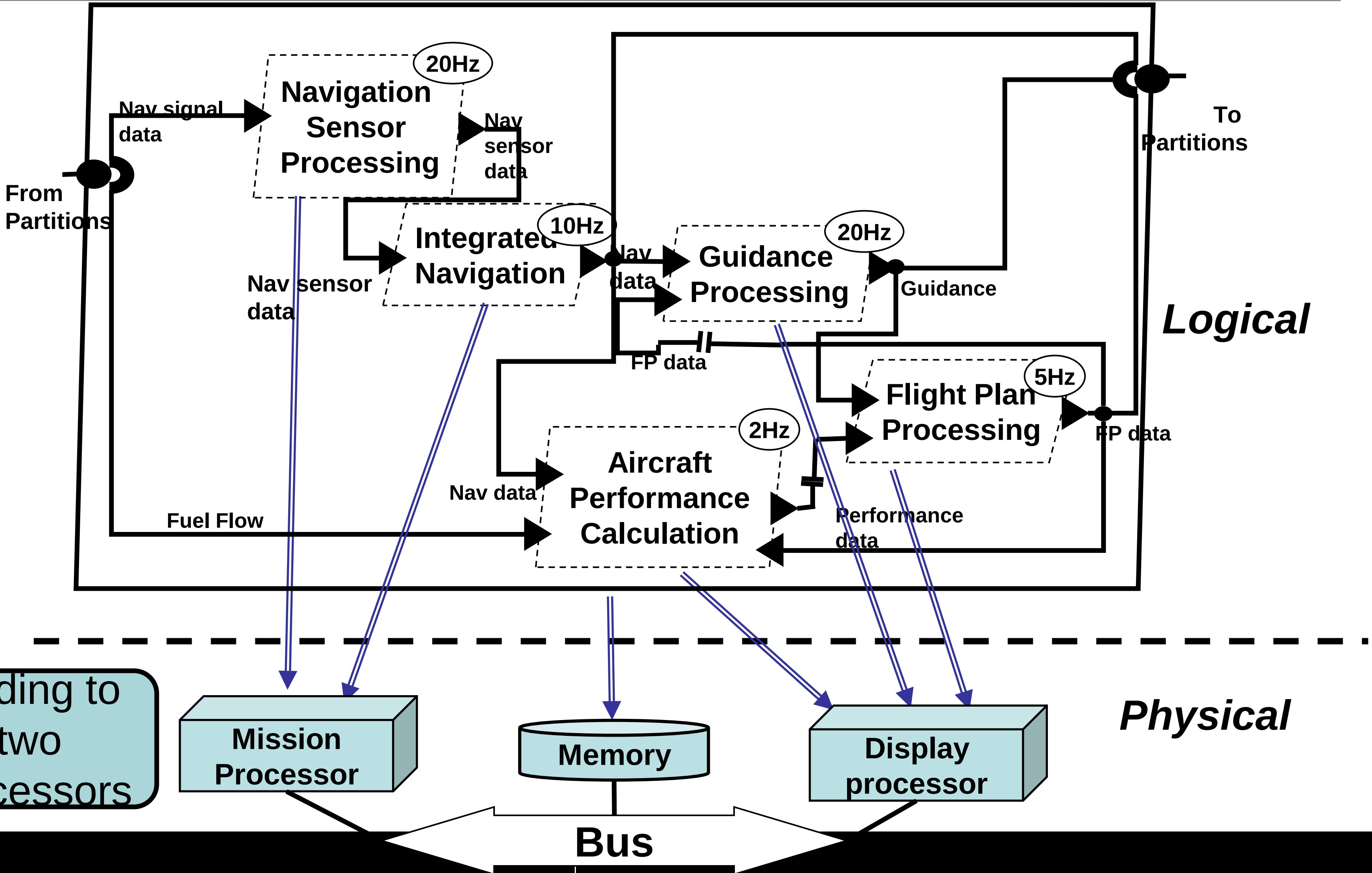
Port group: aggregation of ports into single connection point

Connection: connects ports in the direction of their flow



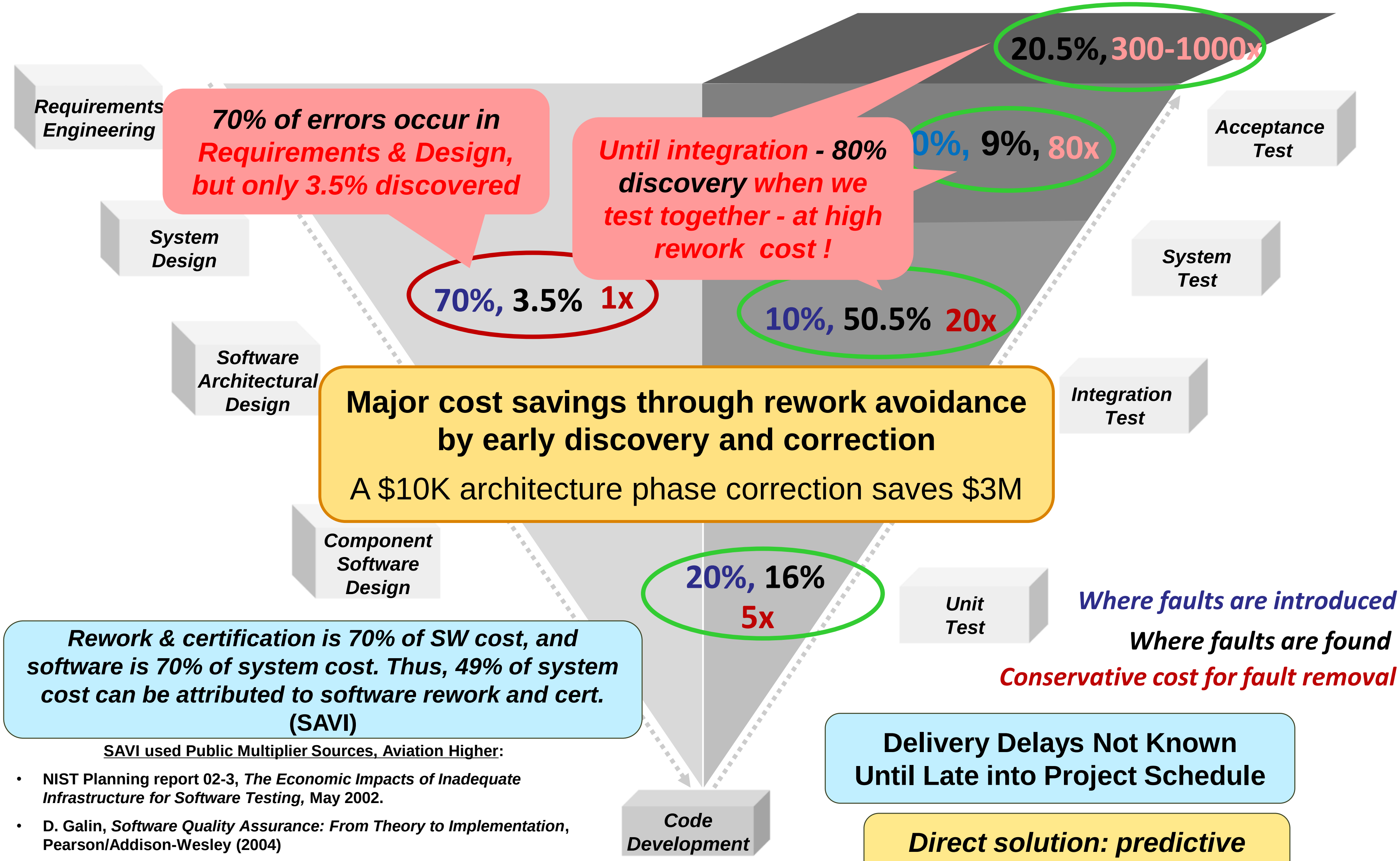

event port connection

Flight Manager Bindings - 2



binding to two processors

UNDERLYING CAUSE – INTERACTION REVEALED
LATE LARGE SOFTWARE REWORK COSTS



70% of errors occur in Requirements & Design, but only 3.5% discovered

Until integration - 80% discovery when we test together - at high rework cost !

Major cost savings through rework avoidance by early discovery and correction

A $10K architecture phase correction saves $3M

Where faults are introduced

Where faults are found

Conservative cost for fault removal

Rework & certification is 70% of SW cost, and software is 70% of system cost. Thus, 49% of system cost can be attributed to software rework and cert. (SAVI)

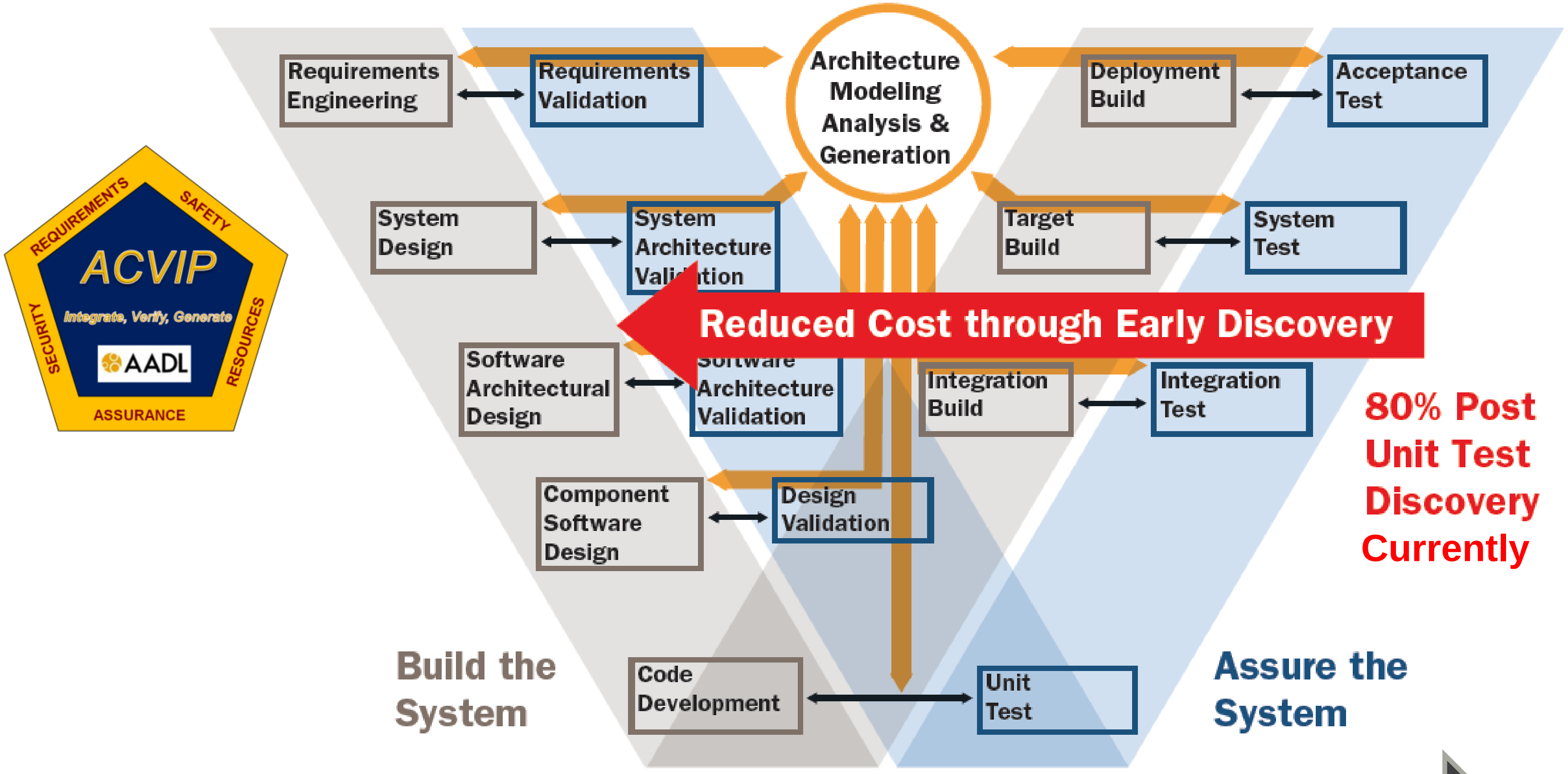
Delivery Delays Not Known Until Late into Project Schedule

Direct solution: predictive system virtual integration!

SAVI used Public Multiplier Sources, Aviation Higher:

- 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)

ACVIP PROCESS APPLIES AADL INCREMENTALLY TO CATCH INTEGRATION ISSUES EARLY



80% Post Unit Test Discovery Currently

Increased Confidence through Continuous Verification and Testing

NEED FOR INTEGRATED ENGINEERING ANALYSIS OF EMBEDDED SOFTWARE SYSTEMS SIMILAR TO PHYSICAL

Virtual Integrated Physical System
Analysis Uses Computer Models (e.g. CAD)

- Aerodynamics
  - Aero elastics
  - Stall and Compressibility
  - Acoustics
- Structures
  - Static and Dynamic
  - Flutter and Vibration
  - Fatigue
- Drive Systems
  - Power Transmission
  - Wear and Fatigue
- Engine
  - Power Available
  - Fuel Required
- Mission Performance
  - Payload
  - Range
  - Speed

Change in # Rotor Blades from 2 to 4

↓

Increased Wt & Change in Vibrations

↓

Increased Power Transmission

↓

Increase in Fuel Flow

↓

Potential Loss of Capability

Virtual Integrated Software System
Analysis Uses AADL Model

- Security
  - Intrusion
  - Integrity
  - Confidentiality
- Resource Consumption
  - Bandwidth
  - CPU Time
  - Power Consumption
- Real-Time Performance
  - Execution Time / Deadline
  - Deadlock / Starvation
  - Latency
- Data Quality
  - Data Precision / Accuracy
  - Temporal Correctness
  - Confidence
- Safety and Reliability
  - MTBF
  - FMEA
  - Hazard Analysis

Change of Encryption From 128 bit to 256 bit

↓

Higher CPU demand

↓

Increased latency

↓

Affects temporal correctness

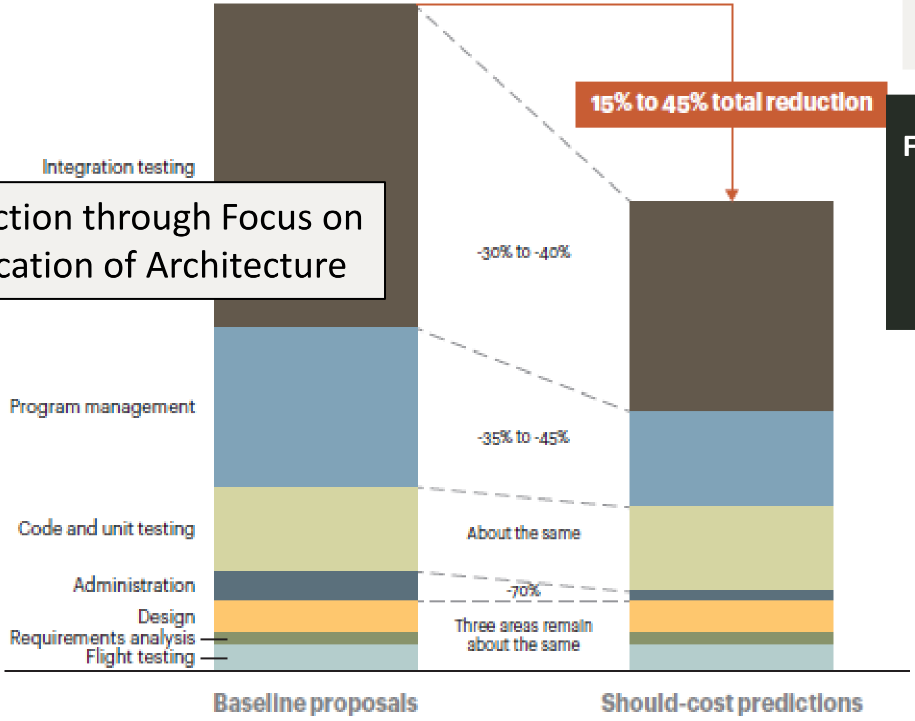
↓

Potential new hazard

Auto code generation from AADL Virtual Model is similar to Automated fabrication from CAD Virtual Model

COST REDUCTION POTENTIAL THROUGH VIRTUAL INTEGRATION OF EMBEDDED SOFTWARE SYSTEMS

ROI on AADL Pilot



Reduction through Focus on Verification of Architecture

ATKearney "Software: The Brains Behind U.S. Defenses Systems"

For a 27.5 MSLOC system Nominal development cost reduction of 26%
(e.g., $2.4B out of $9.2B on a projected 2024 program)

Source: *ROI Analysis of the System Architecture Virtual Integration Initiative* CMU/SEI-2018-TR-002

Summary

- AADL embedded system engineering benefits
  - Analyzable models drive development from requirements
  - Prediction of runtime characteristics at incremental fidelity levels
  - Ability to see side effects of change across RT architecture
  - Design tradespace analysis can be (and has been) automated
  - Critical design decisions are made explicit for reuse/update
  - Predictive analysis of runtime performance/effects early and throughout lifecycle greatly reduces integration and maintenance cost/risk/time
  - Early prototyping or trusted build through generative integration of components
  - Supports integration of multiple domains of analysis for RT systems on a common model with standard semantics and properties
  - Being developed to provide RT analysis capabilities for SysMLv2