

# AADL v3 Roadmap

Peter Feiler

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Copyright 2018 Carnegie Mellon University. All Rights Reserved.

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.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM18-0072

# Overall Strategy

## AADL V2.2

- New AADL V2.2 errata: <https://github.com/saeaadl/aadlv2.2>
- OSATE issue reports: <https://github.com/osate>

## AADL V3

- Working slides & documents
- Prototype implementation
- AADL V3 Issues: <https://github.com/saeaadl/aadlv3>
- Discussion/working document area: <https://github.com/saeaadl/aadlv3/wiki> and committee area at [www.sae.org](http://www.sae.org)

# Roadmap – Active

## Compositional Interfaces \*

- Interface composition, Feature group improvements, Interface properties
- Revised: alternative syntax for named interface composition
- Action:

## Configuration and Choice points \*

- Freezing of design space and parameterized configurations
- Implementation selection for subcomponents, properties, bindings, relation to extends/prototypes
- Revised:
  - Configuration specification, parameterized configurations
  - Unnamed composition
  - Alignment with core syntax
- Action:

# Roadmap – Active

## General binding concept \*

- Binding type, binding point, binding instances (single target, alternative targets), Binding constraints, resources, Resources and resource types, Non-resource binding types, Multiplicity handling, binding of connections to platform flows, binding of features to platform layer
- Open issues: Binding & Arrays

## Array support revisited

- Exposure of index dimensions/sizes in interface via feature arrays
- Connection declarations with embedded index specification
- Configuration of dimension sizes
- Action:

# Roadmap - Candidates

Nested processors, Virtual memory, platforms (Peter, Alexey, Denis, Jerome)

- settled

Virtual platform modeling

- Connections between virtual bus, virtual processor, virtual memory
- Virtual process/memory via virtual bus?
- Mixture of virtual and physical?
- Virtual platform flows

“Hardware” & virtual platform flows

- Flows between platform components
- Flow specs on hardware components
- Target of connection, virtual bus bindings

Virtual devices (Bren)

- What is the problem we are addressing:
  - Device as VHDL and SW device drivers
  - Device as part of the system architecture & part of functional architecture

# Roadmap - Candidates

Unification of type systems and expression languages \*

- Data types, property types, constraint language variable types
- Lists & sets for properties: Set with unique element semantics?
- Union of types: collapse entry point properties (3-to-1)
- *Removal of classifier/reference in expression part (typed expressions)*
- Handling of units: part of value, association via property

Property sublanguage \*

- Properties presented as separate sublanguage from core AADL
- Integration of proposed Units system (ISO, SysML)
- Nested naming of property sets and possibility of inheritance
- Property value: Single assignment, statically scoped default with override
- Stereotype concept

# New Revision Candidates

Flow trees and graphs

Flow categories: sampling, message based

- Sampling flows including up/down sampling semantics and appropriate queuing

Generic ports and other feature categories \*

- Ports without sampling, message (sampling/queuing) distinction
  - Note: down sampling may imply multi-element buffer
- Physical features
- Observable features

Alignment of spec sheet and blue print with component type and implementation

- Currently implementations represent both a variant and a realization
  - Component implementation present in both public and private package section
- Variant characteristics to choose variant
- Realization to expand instantiation



# More Candidates

Interrupt handler (Jerome)

Data aggregation via protocol

Data mapping via new binding/mapping concept

Clean up directionality of access features (Peter) [Errata] \*

- Need for Access\_Right?

Categories on connections: make them optional or leave out? [errata]

Abstract feature as generic feature only

- Not refinable into other feature category

Abstract as generic component only

- Not refinable into other component category

Usefulness of public/private package sections (Bren, Jerome)

AADL\_Project propertyset & project structure (Jerome, Pierre)

Multiple (Mode) state machines aka state variables (Peter, Alex, Bren\*, Jerome, Brian)

- Modes, BA states, EM states, hybrid annex, interacting state machines