

AADL v3 Roadmap

Peter Feiler

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Copyright 2019 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.

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

DM19-0166

Overall Strategy

AADL V2.2

- New AADL V2.2 errata: <https://github.com/saeaadl/aadlv2.2>
- OSATE issue reports: <https://github.com/osate>
- Long term support (LTS) for OSATE 2.x

AADL V3

- Working slides
 - <https://github.com/saeaadl/aadlv3/tree/master/SAEAADLV3>
 - Issues: <https://github.com/saeaadl/aadlv3/issues>
- New draft standard document
 - Document conversion into Restructured Text (RST) in progress
 - Document split into sections
 - Revision of packages, component interface, implementation, sucomponent, configuration
- Prototype implementation started
 - <https://github.com/saeaadl/Aadlv3Prototype>

Migration Path to V3

Instance model representation with minimal changes

- Most analyses operate on instance model
- Documented API

Declarative model

- Translation from V2.2 to V3

Key V3 Changes

Packages and General Syntax

- Import of namespaces
- Property definitions in packages
- Private classifiers and property definitions
- Simpler syntax: no section keywords, no matching end identifier
- case sensitive

Composition of Component Interfaces aka. component type

- *Extends* of multiple interfaces
- Interface without category
- Eliminates need for feature group type

Configuration Specification

- Finalize design
- Configuration assignment of subcomponents with implementation, features with classifier/type (Replaces *refined to*)
- Assign final property values to any model element
- Annotate with bindings, annexes, flows
- Configurations are composable
- Parameterized configuration limits choice points (Replaces *V2 prototype*)

Key V3 Changes

Unified type system

- Single type system for properties and data types
- Records, lists, sets, maps, unions
- International System of Units

Properties

- Stereotypes to specify applicability
- Simplified property value assignment (default, final, override)

Explicit deployment binding concept

- Binding points and binding declarations
- Resource types associated with binding points

Virtual platform support

- Virtual memory
- Connectivity between virtual bus, processor, memory

Flows

- (virtual) platform flows
- Flow merge points

Nested component declarations

- Define nested components without explicit classifier

Key V3 Changes

Array support revision

- Exposure of index dimensions/sizes in interface

Connections

- Distinguish feature mappings
- Reach down of connection declarations
 - Into named interfaces (aka feature groups)
 - Into subcomponent hierarchy

No more category refinement

- Abstract component to other component
- Abstract feature to other features

Modes