

# AADL v3 Roadmap

## AADL meeting Jan 2016

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Peter Feiler



# Overall Strategy

## Major issues

- Require multiple meetings to discuss
- Need to ensure consistency with rest of core AADL

## Smaller issues/errata

- V2.1 errata to go into current OSATE release stream
- Those addressed as V3 change

## From white paper/slides to draft standard

- Incremental prototyping of V3 implementation
- Draft text placement into possibly new document structure

## Compatibility

- Deprecation
- Migration



# V3 Prototyping

Separate from OSATE v2.1 release stream

- New file extension aadlv3

Prototyping schedule and priorities

- End user need
- Validation of new concepts

Meta model changes

- Meta model for instance has not changed much from V1 to V2
  - Is not expected to change much for V3
  - Possible simplification of connection instance support
  - Scalability of generated system operation modes
- Meta model for declarative model
  - Current Meta model size/complexity
  - Impact of Meta model changes



# OSATE Infrastructure Cleanup

- V1 legacy code (e.g., AObject, Location, aadl/aaxl files)
- Error reporting/diagnostics
- Plugin support: run & run as job, read-only/read-write
- Consolidation of public API libraries
- Textual instance model representation
  - Compact & readable
  - Similar to declarative model

Instantiation of any classifier, i.e., also types.



# Roadmap – Active

## Compositional Interfaces (Julien, Peter\*, Alexey\*, Jerome, Bren)

- Interface composition, Feature group improvements, Interface properties
- Action: name disambiguation, shared features
- Schedule:

## Variability points and configuration (Peter)

- Implementation selection for subcomponents, properties, in modes mappings, bindings, prototypes: reach down into model
- Syntactic distinction of architecture design and configuration, **final** design/configuration
- Action: configuration of virtual bus, virtual processor, and other implementations (Implemented as)
- Schedule:



# Roadmap – Active

## General binding concept

- Binding type, binding point, binding instances (single target, alternative targets, target sequences), Binding constraints
- Action: configuration assumptions, binding sequences and flows
- Schedule:

## Array indexed connection declarations (Peter, Brendan)

- Instance configuration: pattern or index mapping set on top connection
- Exposure of index dimensions/sizes via feature arrays
- Feature mapping connection, index subset mapping
- Action: interplay with configuration view (mapping) – inverse of mapping
- Schedule:



# Roadmap - Candidates

Nested processors (Julien, Jerome, Alexey)

Virtual memory (Alexey, Julien)

“Hardware” flows

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

Virtual platform connectivity

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



# Roadmap - Candidates

Unification of type systems and expression languages (Alexey, Brian, Serban)

- Data types, property types, constraint language variable types
- Lists & sets for properties: Set with unique element semantics?
- Union of values: 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
- Nested naming of property sets and possibility of inheritance

Property inheritance, override, and **final**





# Inheritance & Overriding of Property Values

Property associations without applies to

- Applies to was originally introduced to deal with model reference values for expressing bindings
- Property value rules allow overriding from outside in
- Used to allow configuration of data sets to instance model
- Configurable data set concept introduced in V2

**Final** on property association vs. final as default

- Override in extends hierarchy (always unless **constant**)
- Override in component hierarchy (definition states **inherit**)
- We have **constant** instead of **final**
- Final as default
  - Explicit specification of variability point
  - Inherit without override

Default value for properties at definition time

- Examine each for actual need
- Default with component classifier; is inherited



# Roadmap - Candidates

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

AADL Project (Jerome)

Multiple (Mode) state machines aka state variables (Peter, Alex, Bren\*, Jerome, Julien\*, Brian) [kickoff in Feb]

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



# More Candidates

## Virtual devices (Bren)

- Device as VHDL and SW device drivers
- Device as part of the system architecture & device as part of functional architecture

## Flow related

- End to end flow requirement without an implementation (Approved errata)
- Flow path/source/sink as end to end instance



# Add'l Candidates

Table of Content (Jerome, Peter)

Core constraints by constraint expression rather than properties

Applies to in property definition: do not require listing of enclosing component categories when property is inherit

Call sequence is currently not a name space for call identifiers.

Interrupt handler (Jerome)



# More Candidates

Data aggregation via protocol

Data mapping via new binding/mapping concept

Clean up directionality of access features (Peter)

- Need for [Access\\_Right?](#)

Categories on connections: make them optional or leave out?



# Table of Content: Organization of Std Document

## Currently

- Generic package and component concepts
  - Includes annex mechanism, prototypes, aliases, abstract components as category
- Specific component categories
- Generic concept of features and specific feature categories
- Generic concept of connections and specific connection categories
- Concept of flows
- Property sublanguage
  - Property association, property and type definition
- Concept of modes
  - Impact on architecture hierarchy & topology
- Concept of system instance



# Table of Content: Organization of Std Document

## Proposal

- Generic architecture concepts
  - Components, features, connections/flows
  - Modes, configurations, instances
- Specific component categories
- Property sublanguage
- Model organization concepts
  - Packages, aliases
  - Annexes, data sets

