Notes from

AADL standards meeting Sept 21-24 2015

* Location Seattle, USA. Sheraton Seattle Hotel – 1400 Sixth Avenue – Seattle, WA98101 USA
* Meeting Room 213 at the convention center

Local attendees: Brendan Hall, Peter Feiler, Greg Newman, Bruce Lewis, Brian Larson, Etienne Borde, Pierre Dissaux, Alexey Khoroshilov, Serban Gheorge, Huafeng Yu, Julien Delange, Jerome Hugues

Remote attendees: Myron Hecht, Dominique Blouin, Jean-Pierre Talpin, John Glowa, Kevin Herring

# Monday, Sept 21

* 0900-0930: AADL standardization committee news + action items (Bruce Lewis)
  + Chattanoga – for Spring Meeting. But there may be an AADL workshop in Austin for research interests.
* 0930-1000: OSATE update (Julien Delange)
  + ICSE in Austin, Compac is in April
  + User group orientation – do we add a day to the meeting? Go to JMR meetings. Do virtual meetings? How do we form a community of Interest?
* 1000-1030: AADL v3 roadmap review
  + White paper needed to remove Public-Private to describe the use of repositories to achieve privacy.
* Discuss about unification of type systems
* Useful of private/public sections
* Multiple mode state machines aka state variables - modes, BA states, EM states, hybrid annex, interacting state machines
* Virtual devices?
* Flows: need to have an end to end flow requirement without having the implementation
* Hardware flow
* Nested property naming
* Textual syntax for instance model
* White paper needed to remove Public-Private to describe the use of repositories to achieve privacy – Peter Feiler
* 1030-1100: break
* 1100-1200: AADL v3 Configuration & Binding
  + Configuration via a subsection called configuration
  + When am I just refining vs. extension
  + Configuration allows us to reach down to define more completely.
  + Have a type, configure in an implementation?
  + Could change classifier – abstract to system, etc.
  + Could develop a full set of types then configure implementations.
  + Brian – disagrees - configuration should be disjoint from the architecture.
  + Here configuration is being done in the architecture.
  + Julien – we have configuration now, refines to does configuration to a subcomponent.
  + Peter – all we are doing is providing blueprints where they do not exist or not fully define.
  + This is configuring the instance model. Allows to reach down from the top.
  + Using configuration to describe what the instance model looks like.
  + Brian is doing something separate from AADL for configuration but this allows a more organized approach within the AADL.
  + Summary – refines restricts the design space, notationally we use extends notion to do also refinement. Could include a section for Configuration and refine from this definition. Could be configures as well as extends. Provider makes decisions about the instances but leaves some open for the user to define. Can we stay within the existing language as much as possible (use extends in both cases) , or just use extends for extending types and implementations and configures for refinements.
  + Decided: Separate construct for refinement agreed to .
* 1200-1300: Lunch
* 1300-1400: AADL v3 Configuration & Binding
  + Bindings and other Mappings between System Hierarchies
  + Do we need to go to the top to see both or can we develop another approach.
  + In some cases we do not have to follow the hierarchy.
  + Currently done with properties.
  + Bindings are not distinguishable from others, we need an explicit concept rather than hiding in the properties.
  + The binding instance is the place where it is bound.
  + system configuration top.configured extends top.basic
  + system top.configured configures top.basic
  + configuration subclause configures subcomponent (only refined to)
  + the key issue is to bypass the hierarchy and being able to reach down the components hierarchy
  + minimum change required: reach down the component hierarchy for all potential refined to statement
  + Brian: we mix architecture and configuration - need to clarify the concepts and have configuration in a text file
  + Brian: ok, part of the configuration is done in AADL
  + Peter: one of the biggest issue was the prototype mechanism introduce many issues
  + preconfiguration vs. configuration - where should we finally define the configuration?
  + Decided: Separate construct from refinement was agreed to.
  + Mapping are expressed by properties today
  + Binding instance: deployment configuration - what we have right now
  + Binding points
  + Binding type: define the type of bindings we can have
  + new binding section in the component implementation
  + abstract component used as feature groups
  + able to connect all the interfaces in a components by using directly the feature groups
  + Binding type
    - Binding ProcessorBinding, MultiCoreBinding, ProtocolBinding suggested. But others are allowed, there would be pre-declared. New ones could be developed.
    - Binding ProcessorBinding ; {thread, thread\_group, system} -> {processor, virtual processor, system}
    - Binding Instances
      * ProcessorBinding : App.sub.proc.thread -> platform.node.cpu1;
    - Binding point
      * **Virtual bus** protol
      * **Features**
      * requiredProtocols: **Requires** Protocolbinding [ VBOrBusClassifier ];
      * ProvidedService: **Provides** ProtocolBinding [ VBOrBusClassifer];

Avoids going up and down the hierarchy.

* 1430-1600: Behavior annex Errata (Etienne Borde)
  + Informal ballot document provided. Covering each accepted errata.
  + Fixed D.4-06 – was missing separators
  + Fixed D.5-02/D.5-03 – this significantly reduces model checking time.
    - Depends on an update of the core, this core update is required.
  + Fixed D.5-04/D.5-06 Need synchronization iwht data modeling annex and core. When the name in the BA is the same as a mode, then they refer to the same mode. BA gives you a way to express more detail on the transition logic. Treated in Wiki.
  + D.7-04 fixed.
  + These modifications have been added to the document sent to you last week.
  + **Action**: Everyone - comments to Etienne by the middle of Nov. Pierre and others will test the syntax after mid November. Changes from this meeting will be added to the document by Etienne. Etienne will provide changes we approve by the end of the week.
  + BLESS assertions now expressed a properties.
  + D.7-08 Rem/mod difference – definition from Ada. Accepted.
  + D.7-09 Add the possibility to test if a data is not initialized. – uninitialized becomes an new key word. Need to have a default value to avoid uninitialized. Could use zero for some data types. Solution: Add a legality rule : data used on the Right Hand Side of any expression in BA must have been initialized. Accepted.
  + D.7-10 – ability to define a behavior variable as a “final” variable; it can only be initialized once. Same as a final variable in JAVA. Remove until we find the requester. Alexey will check. Does not seem to be needed. Must staticly check. Could be discussed in the Data Modeling Annex. See notes from Etienne for more detail.
  + D.7-11 Add the possibility to define a behavior variable as a “constant”. Initial value is defined staticaly using the Data Modeling Annex. Must be initialized on declaration. Accepted. Communicate with Jerome on the Data Modeling Annex.
  + D.3-08 reject – savings are small.
  + D.4-05 Support for internal features – one annex interacting with another annex. The feature is an internal feature. You want to prove that the failure behavior or the recovery is sufficient. Allows the behavior to react to a failure via the error annex. Some logic in the BA to handle. To allow a tighter argument. Transitions can be triggered by internal events, the component can react in an active state. Should be used in an abstraction (abstract spec) rather than in an implementation since the interaction has to be more specific. How do you respond to this event? Is it sufficient for recovery?
  + It would be in the BA of the component that senses the error and raises it to be handled. It comes from an error defined in the Error Annex. Error model documents what is detected. Behavior model handles. Would provide an example in the Behavior Annex. **Action:** Etienne (with Peter and others) will develop an example). Define the semantics of this internal event and describe explicitly the reaction.
  + D.6-12 exception management - need input from Denis. How should we show where we go on the exception. Using the left side of the example only. Worked out a way to show action. Action: develop solution from meeting discussion and talk with Denis.
  + D.7-07 Inside computation – parameterize time. Current BA uses only integer values. Not ready for acceptance, needs more understanding of how it will be used.
  + BLESS assertions now expressed a properties.
  + D.7-08 Rem/mod difference – definition from Ada. Accepted.
  + D.7-09 Add the possibility to test if a data is not initialized. – uninitialized becomes an new key word. Need to have a default value to avoid uninitialized. Could use zero for some data types. Solution: Add a legality rule : data used on the Right Hand Side of any expression in BA must have been initialized. Accepted.
  + D.7-10 – ability to define a behavior variable as a “final” variable; it can only be initialized once. Same as a final variable in JAVA. Remove until we find the requester. Alexey will check. Does not seem to be needed. Must staticly check. Could be discussed in the Data Modeling Annex. See notes from Etienne for more detail.
  + D.7-11 Add the possibility to define a behavior variable as a “constant”. Initial value is defined staticaly using the Data Modeling Annex. Must be initialized on declaration. Accepted. Communicate with Jerome on the Data Modeling Annex.
  + D.3-08 reject – savings are small.
  + D.4-05 Support for internal features – one annex interacting with another annex. The feature is an internal feature. You want to prove that the failure behavior or the recovery is sufficient. Allows the behavior to react to a failure via the error annex. Some logic in the BA to handle. To allow a tighter argument. Transitions can be triggered by internal events, the component can react in an active state. Should be used in an abstraction (abstract spec) rather than in an implementation since the interaction has to be more specific. How do you respond to this event? Is it sufficient for recovery?
  + It would be in the BA of the component that senses the error and raises it to be handled. It comes from an error defined in the Error Annex. Error model documents what is detected. Behavior model handles. Would provide an example in the Behavior Annex.
  + **Action:** Etienne (with Peter and others) will develop an example). Define the semantics of this internal event and describe explicitly the reaction.
  + D.6-12 exception management - need input from Denis. How should we show where we go on the exception. Using the left side of the example only. Worked out a way to show action. Action: develop solution from meeting discussion and talk with Denis.
  + D.7-07 Inside computation – parameterize time. Current BA uses only integer values. Not ready for acceptance, needs more understanding of how it will be used.
* 1600-1700: Behavior Annex with BLESS/Sync Integration (Brian Larson, Etienne Borde, Jean-Pierre Talpin)
  + Overran time – summary later.

# Tuesday, Sept 22

* 0900-1030: AADL v3 Compositional Interfaces
  + Builds on extend to allow two types to be integrated into a third type
  + Not for implementations with extends for two types.
  + May be limited to abstract to reduce the number if issues. Two abstracts become a component type.
  + Should we allow any category?
  + Initially one should use abstracts
  + Or could only bring the interface.
  + Composition of interfaces impacts Inverse Of, no inverse for the physical interface since everyone requires access.
  + What about overlap, they must name match to be understood to be the same.
  + Disambiguate name conflict: qualify with interface name. Do they represent the same feature?- deal within the same way for feature groups.
  + Keep the interfaces flat, no hierarchy
  + Connecting all features of an interface, Features can be reference individually
  + Serban – what about allowing package\_name::feature group, not in the language but perhaps easier to deal with than alias.
  + Disambiguation of names is the next
  + Brian – is there a way to do these things in what we have now, why change? Feature groups do 95 percent – Serban
  + Feature groups could be changed – the only issue would be in properties, Alexey wants more natural expression. Serban and Brian favor staying closer to the current language. Bruce – need to examine both. Want to minimize change.
  + Composition of Modes next
  + Inline mapping can be done today with feature groups. The only thing we are gaining is in one case.
  + Brendan – tagging private or frozen. Issues in feature group today. But I like the interface approach. Brendan will review the slides.
  + Mapping may resolve the problems with feature groups. John Glowa
* 1030-1100: break
* 1100-1130 664 Analysis with Bridge from 653 to 664 to enable optimization and reliability analysis (Etienne Borde)
* 1130-1200 FASTAR – AADL toolset for Timing Analysis Integration (Steve Vestal)
* 1200-1300: Lunch
* 1300-1400: Network Annex Update (Alexey Khoroshilov, Tiyam Robati, Brendan Hall)
  + Latenyc (timing) analysis
  + Buffer capacity
  + Consistency
  + Syntheis and trade-off
  + Safety
  + Use of specific property
  + Capture TTEthernet and AFDX
  + Current status: AFDX way more advanced than the TTEthernet. Still a lot of thigns to clarify for TTEthernet
  + Brendan Hall will then join the meeting tomorrow to talk about TTethernet
  + Goals of modeling annex: configuration generation, timing analysis, latency analysis
  + …
  + Current status – how we propose to model
    - Property set with constraints
    - Already in MASIW, most of them
    - Processors communicating Avionics Network
    - Use a virtual bus
    - Then add physicial model
    - Bind virtual bus to processors, the simplest approach
    - Add avionics switches
    - Use connection binding to physical
    - This can be used for synthesis
    - Latency constraints, other constraints added
    - Then add virtual links as virtual buses in the virtual bus and use connection bindings
    - Using AADL properties to build a route table.
    - Very good for symmetrical
    - Future binding points would help to replace virtual bus.
    - Jerome – we need a way to inherit many of the properties that get repeated on each virtual link
    - AFDX reference models, can send
    - But we do not have an example of redundancy, network A and network B.
    - TTEthernet has virtual frames and rate constraint frames, have the properties, but only about 50% complete. She will send an updated version soon.
    - TTEFrame across Synchronization Domians
    - RCFrame vs. AFDX VL
    - Redundancy (network A/B)
    - Annex should be done soon if possible
    - Need to describe modeling of Messages, could be a generic section of the annex.
    - We should support code generation – bind on the sampling port, configure tables. Set down with Jerome.
    - AADL core development plan – some parts of AADL V3 will simplify.
    - For safety, we should talk about end to end performance.
* 1400-1500: AADL V2.1 Errata
  + Virtual Channels, Flows, and Bindings
  + Hardware Flows
  + Issue of copy/paste for networks
  + Composing flows out of shorter flows, here’s my branch point.
  + Instantiation of Flow Paths –
  + New Properties
    - Thread duration – Thread ports to complement completion (final). Application robotics.
      * What is the time reference. How should we do this type of modeling. Time reference point, duration on modes vs. duration on threads.
    - Need white paper to exercise the example of expanding a design beyond internal feature
    - Property Association Related (V3 consideration)
    - Virtual Memory through a virtual processor.
    - Input to Output Connections.
* 1500-1530: Break
* 1533-1700: Attend ATC 403 session with AADL presentations

# Wednesday, Sept 23

* 0900-1030: AADL v3 discussions (Nested processors, virtual memory&memory configurations)
  + Questions on the integration of BA and Synchronous, BLESS.
  + Plans to build a toolset at Renne, scheduler and generator, working on how to do it over the next two years.
  + Quegen – reason about multi-core systems with synch
  + Missing elements relative to integrating a family of specifications
  + To do – What is the objective of the formalization of BA
  + Formalization includes support of sych, async and timed systems as well as logic checked.
  + Bridge to Polychrony
  + Etienne – keeping the skeleton large and having a demonstration of it.
  + JP – have a first draft of BA annex with synch? Etienne - Continue working with BA to keep the skeleton and demonstrate it can be used. Brian – willing to work on it together, POP.
  + Both – API of the BA, but also work on integration of synchronous/reative subset/part. Usage should be explained to the committee.
  + **Decision** – we will work on upgrade of the BA in the normal manner with errata as the means of update. Short term - skeleton for BA, Longer term – merged semantics longer term but also implementation oriented work. Formalization of the current BA, then
  + abstraction properties and data for the analyses. Example Ravenscar. Then the properties that do the scheduling that can be done in that context.
* Nested processors, virtual memory
  + We have some ideas but now have questions.
  + Should we use memory to model logical components – a hard drive. Don’t use a device to model a hard drive, its just memory. But Brendan – we have file systems in a device.
  + Virtual memory abstraction – it have to become something, a binding. See binding points. We can attach properties to binding points. A binding point is like a port. It tells you how it relates to the layer below. I am an entity that needs to be bound (thread to a memory, processor). Requires access is at the same level. This is below.
  + Brendan – do we need a virtual protocol. We use a virtual bus for several purposes. **Action:** Peter request to all - Given the notion of virtual bus it contains the idea of a channel and a protocol, please provide the differences, why we might need two concepts.
  + Peter, do we need to have in the processor two types of memory – for O/S, and for applications, then applications segmented by OS.
  + See Julien’s targeting of Deos and VxWorks.
  + Using Abstract to model things like power, etc.
  + It would be good to have libraries for things like power, hydraulics, etc . A pre-defined set of concepts for capturing non-software/hardware analysis.

* 1030-1100: break
* 1100-1200: AADL v3 discussions (array connections, unified type systems)
  + Should we allow array specification at multiple levels?
  + We do now but its complex.
  + Nested arrays require exposing multiple levels.
* 1200-1300: Lunch
* 1300-1400: AADL v3 discussions (array connections, unified type systems)
  + We have multi-dimensional arrays for components, Single dimension for features, arrays at different levels of the component hierarchy
  + Proposal to use arrays and be able to uses indexes
* 1400-1430: Break
* 1430-1630: Expression language and Constraints Annex Draft plus Update (Serban Gheorghe)
  + Lots of common ground with Resolute
  + Comparison of different existing constraint language
  + Evaluation of resolute
  + Could we build on it?
  + Constraint Language MSC language, how should we build it.
  + Evaluated Element types hierarchy, Basic types, Access methods, Computational functions and expressions, Theorems or claim functions, check/prove statements
  + Proof execution units - MCS has but Resolute not. Resolute groups single claim functions.
  + Scalability – without modifying model is in MCS, this is industrial. External control.
  + Resolute does all its verification on the instance.
  + Verification on the declarative model is not sufficient, it’s the instance model that resolves the context.
  + Need a phone conference to resolve the next step. Appears that there should be two languages. One related to what verification activities should we have. The second one to walk AADL models to verify structural and computational part, how do I get hold of model elements, how is it usable, how do we add new capabilities, types, annexes. How do we add the behavior part. Resolute has verification plans, that should be extended. But we should separate the language. If we need to have an expression language different places its definitely better to have one that is standard (Alexey). Serban, Brendan, Jerome.
  + Brendan – we need both the standard constraint analysis specification as well as the plan to verify (harness that can call many forms of evidence of satisfaction) and verification mechanism which in the AADL is an enhanced Resolute, verification of constraints.
  + Verification Activities, Compositional verification plans, Verification Method Registry.
  + Yield Assurance results
* 1630-1700: Joint Common Architecture ACVIP Shadow Report (Alex Boydston, Bruce Lewis)

# Thursday, Sept 24

* 0900-1030: AADL v3 discussions (TBD) (Peter Feiler)
  + Derive an architecture that is more amenable to change. Brendan, we have DARPA project.
  + BRASS – Honeywell working with SRI, Kick-off
  + Don’t want to embed requirements into the AADL model but must link to the model. At each level you need to have a verification plan based on the requirements updated for the level of the model (derived, etc). The assurance case is based on all three, the models, verification plan, and requirements. Slide 10
  + Residual risk – what is left when my hazard mitigators are considered. Brian
  + What type of syntax do I need to walk AADL models, other types of models.
  + Standardization of a grammar – agreed, we need. Processing of the grammar, some experimentation needed.

* 1030-1100: Break
* 1100-1130: AADL v3 discussions wrap-up and follow-up (Peter Feiler)
  + See Roadmap slide
  + **Action:** Brenden : Webex with Peter, Pierre to explore more the array solutions suggested.
  + **Action**: Peter: Determine if we can do state machines aka state variables at the next meeting.
  + Brian we need to evaluate automated unit conversion (but that takes funding).
  + We need that common grammar for constraint language, this is not an expression language, its types and units. Peter has explored. Brian suggestion.
  + Peter – we should not do a total new metamodel for this area.
* 11:30-1200: Planning for next meeting (Bruce Lewis)