AADL standards meeting Jan 30-Feb2, 2017

* Location Toulouse, France.
	+ Meeting information
		- At the IRIT, Institut de Recherche en Informatique de Toulouse – CNRS
		- Rue Charles Camichel, 31000, Toulouse
	+ <http://maps.google.com/maps?q=Rue+Charles+Camichel,+31000+Toulouse,+France&z=16>
	+ Please send Marc your name, nationality, current residence and email so he will have you on the list to enter the campus.

# Monday, Jan 30

* 0900-1000: AADL standardization committee news + action items (Bruce Lewis)
* 1000-1030: AADL v3 roadmap review (Peter Feiler)
* 1030-1100: break
* 1100-1230: AADL v3 discussions (Nested processors, virtual memory & memory configurations) (Peter Feiler, Alexey Khoroshilov, Jerome Hugues)
* 1230-1400: Lunch
* 1400-1500: INGEQUIP project feedback for V3, Capella to AADL transformation (INGEQUIP Team)
	+ Dr. Eric Jenn presenter
	+ CAPHA the next project and will also use AADL
	+ Thales leading (Eric from Thales).
	+ New to AADL
	+ Spent about 2 months on AADL, on equipment engineering, engineering in the large, not domain specific.
	+ Critical Applications on Predictable High-performance Computing Architectures
	+ Design space exploration
	+ co-design
	+ formal development methods with FIACRE with the idea for going up to AADL
	+ Partners – Airbus, Airbus with SAGEM with Thales in aviation domain, ….
	+ Created a Robot as a demonstrator for the technologies, runs on a taxi way on an airport avoiding collisions, a system of robots.
	+ Embedded part – Mission system and Power management, using CAN bus
	+ Mission system is the brain of the robot. Like in an aircraft with left and right channels.
	+ COM MON architecture for safety
	+ MON on a ZYNQ board with multi-core processor and custom co--processor and COM on a Power PC
	+ Power system on a PPC.
	+ Not all the functions fully implemented but we are still working on.
	+ Using AADL, Capella, Matlab, OMEeT++, Fiacre, …, AADL must live with these.
	+ REX on AADL – AADL used to initialize the HW/SW design, then to refine the design, AADL to explore and optimize, AADL to express design alternatives, AADL to measure and validate feasibility ….
	+ What is the status of the community
	+ Is it deployed on a large scale, why not
	+ What is the long term prospective roadmap
	+ Which tool to do what
	+ Maturity of the tools
	+ Do the tools scale up
	+ How does it compare to other architectural languages
	+ We used AADL to communicate human to human, had some difficulty to get system engineers to use but easy for software engineers
	+ Most models elaborated by newbies with no big difficulty (at least for the software architecture)
	+ Pretty friendly for SW engineers (textual concrete syntax)
	+ Used to generate code skeleton for application code
	+ No support for part of our platform (Trampoline and OSEK)
	+ No generation of OIL configuration files
	+ Conclusions/Questions
		- Maturity/stability of the code generation tool
		- Capability to qualify the code generation tool
		- …: a clock synchronization protocol
		- Achievements – not really completed experiment, but ..
		- Significant effort on the FIACRE side
	+ Generation of the configuration files for virtual platforms
		- Achievements – first space qualified code generation
		- first prototype of AADL to VP for Space Co-design using OCARINA
	+ Formal verification of temporal/timed properties – our use case
	+ From AADL to Space …
	+ Specify and automate Design Space Exploration (DSE)
	+ Achievements – Ad-hoc modeling of multiple design alternatives of an image processing chain, with hard cores, soft cores, FPGA-implemented co-processor
	+ Multi-core – two cores and FPGA on the ZYNQ board
	+ Design alternatives – multi-core alternatives, where executed and how parallelized, alternatives with the FPGA for software or hardware implementation
	+ Used multiple Microblaze, Dual Core Cortex, etc. and evaluated speed up of signal processing
	+ Simulated multiple rovers and their performance
	+ High-level validation of architectural choice regarding temporal properties
	+ Used AADL with Ellidiss simulator to validate schedule
	+ Investigated bandwidth on bus, with the simulation of messages
	+ This is the behavior of hardware, not software. Can we model hardware with AADL? We managed the model to make it fit. We use a thread to represent the hardware. We can attach a BA specification to anything, including hardware.
	+ You could use periodic or asyn dispatch on threads, I can model any behavior because its periodic or sporadic. Just about whatever you want to do in VHDL you could do in the BA but AADL is not set up for calculating some things in VHDL, but that is not the objective, it’s to capture an abstraction of the hardware behavior. (Brian Larson was a VHDL designer for a time and agreed). Question is what we need in an abstract model. I need to make some assumptions on how things are dispatched. We need to model the dispatch and end to end latencies.
	+ High-level validation of architectural choice regarding temporal properties (incl scheduling with HW and SW events
		- Achievements – very simple model of the image processing Chain pipeline, simple model of the buck controller.
	+ AADL for low-level SW and HW/SW modeling
		- Use AADL to account for HW/SW effects, estimate bus loads, CPU loads
		- Important to multi-core
		- Achievements – first models of software down to the hardware such as IP configuration registers (eg Autosar like)
		- Conclusions – requires modeling of HW/SW interactions: modeling of ..
	+ AADL for hardware modeling
		- Modeled DC-DC converter – buck boost converter
		- Modeled the hardware parts and the software parts together
		- HW elements that impact significantly SW performances (interconnections, cache coherency, DMA, caches effect, synchronous and asynchronous dispatch…
		- Had to model hardware connection using buses (a simple wire would be better)
		- Is a hardware block a system or a device – we used system, how should hw behavior be represented (currently a thread).
		- How to insure consistency between the device and its implementation as a system
		- Used the feature groups to model Hardware Register declaration
		- Not done modeling of Bus bridge modeling with parallel access, modeling of effects of pre-fetch from flash on bus loading, modeling of Effects of DMA traffic, modeling of interrupt and reset behavior (is it possible via event port?)
		- CAPHCA – will be much more efficient in AADL since its modeling safety critical systems. “to provide methods and tools to develop safety critical applications using modern, high-performance, multi-core SoCs”
		- Focus – Determinism, Performance
			* Where are the sources of SW/SW, SW/HW non determinism?
			* How to model them?
			* How to mitigate them?
			* How to express and exploit potential parallelism ..
		- Potential role for AADL
			* Model parallel architectues
			* Low-level effects and interference channels
			* Inferference analysis
			* Virtual platform configuration
			* (Support DSE)
			* Model instrumentation (what can be observed)
			* Model stimuli
			* Verify behavioral properties related to concurrency (deadlocks, race conditions, ..)
			* Verify schedulability
			* Past projects did mostly transformations to AADL so we are doing research on AADL itself in the aero and automotive domains (Philipe Cuenot – Continental attended the meeting as well).
			* This is a three year project.
			* Focus on modeling at the appropriate level.
			* Multi-phase development with input to certification authorities, want to show the certification authorities that I have good understanding of the system.
			* We should have done more on what exists in the state of the Art in AADL.
			* Brian – NASA research on Trade-Space Design Tool will be made available. Will be Open Source. Is available today.
* 1500-1530: Approaches to FMI with AADL (Jerome Hugues, Raphael Faudou)
	+ Context of Study
		- AADL has limited capability for physical models, interoperability with simulations
		- AADL has code generation which allows for rapid prototyping
		- Desire connection with other models such as Modelica, both functional and environment.
		- FMI is Functional Mock-up Interface, it’s for system engineering simulation.
		- Two approaches will be supported
			* Model-Exchange – fetching information from each of the models, FMU’s
			* Co-Simulation – this is the one we will primarily investigate, continuous time solvers with AADL
			* Each FMU has it’s own time step
			* Solver tool integrates them. For simple time steps this is easy, like real-time system.
			* Example – Lunar Lander, a case study to drive requirements and the implementation. FMU’s become devices that are dispatched by the AADL.
			* Master schedule is easily defined in AADL like a real time schedule.
			* Need a set of tasks then can run the simulation.
			* Execute and read/exchange the data from the components. To get right must be correct timing for reading.
			* Provides real time simulation with AADL as an FMU
			* Or provides an AADL controlled co-simulation.
			* Extending scope of tools: ANSYS Simplorer, SCADE, AADL Inspector Simulator
			* Will be ready to demonstrate for next meeting. Jun 5 in Atlanta. Update.
			* An opportunity to push improvement for the FMI community as well (power of the controller of the simulation via AADL).
* 1530-1600: Break
* 1600-1700: Network Annex Draft Review and Update (Alexey Khoroshilov, Tiyam Robati, Brendan Hall)
	+ Some progress but not so much
	+ Have some questions I would like to ask
	+ Trademarks – Bruce – we should proceed with the draft and ask SAE to get the permissions. **Action:** Bruce will send an email to Dorothy.
	+ General network properties – probably go in core. Jerome sent the link. Expected latency and max latency, expected latency is a budget, actual latency, max latency related to calculation.
	+ JH- Is it preferred to have a record of related properties?
	+ Network Annex is for 2.2 so V3 capabilities would not be available.
	+ Need a component to be an interface (a Network Interface Unit). Could be hidden or in a separate partition. Entrance point for the bus would be through this partition or component.
	+ Consistency rules – can write with Resolute. Could put in annex with Resolute to be more formal.
	+ Multi-cast – several outputs, can be very difficult to understand. Hardware flows could help here. Goes to switch then several directions.
	+ Idea - Binding software flows to hardware flows, now we have more to work with. Working the idea of tree of flows.
* 1700-1730: Error propagation paths in EMV2 (Alexey Khoroshilov, Denis Buzdalov)

# Tuesday, Jan 31

* 0900-1030: AADL v3 discussions (Compositional Interfaces) (Peter Feiler)

Composition of Interfaces

 Before: Locally added feature name cannot exist as a name already in the combined component. Now proposed: Matching features: from Logical and Admin represent same feature. Overlap allowed. Can we have the compiler notify the user that the two share the same feature? Renaming?

 “Inverse of” changed to Reverse. Reverse applies to features that are directional. Can keep physical interfaces separate from logical. That way logical can be reversed. Physical you would not. Acceptable solution.

 If I have two interfaces that are the same – for a voter. Voter taking input from multiple instances of same subsystem.

 Extends means adding things. Refine is to change something. Compose (bring together) then extend, then refine? Configure used later. Will revisit refine?

 **Action**: Peter will put example on the GITHUB

Do we use extends for everything or do we try to separate uses of extends with another keyword.

Nested Interfaces – like nested feature groups. Deprecate feature groups in V3.

Specifying that interfaces are merge-able? Named spaces?

Do we want to restrict more inheritance in the case of implementation extensions?

Composition of Properties – properties section of interface – equivalent of data set (mini) annex. Feature groups had types that had properties that applied to the features in the group

Composition of flows –

Composition of modes – one state machine (mode) per component

Annex composition

Interface as a keyword

Composition of implementations (implementation views) - Power supply view, platform view, physical & digital connectivity views

Renaming to see how we remain compatible

* 1030-1100: break
* 1100-1130: Continued (Compositional Interfaces)
	+ Switch to configuration
	+ Looking at it as a separate concept
		- Subcomponent -> implementation
		- Feature classifiers – Port data types, Access types
		- Property values
		- Multiplicity
		- …
	+ Prototypes we used but had to work up/down the hierarchy
	+ Concept of configured – this part is frozen, structure is final, then user can change the details
	+ Configuration of property values, examples for security, safety, performance
	+ Configures …with, composes?
	+ Parameterized Configuration – kind of equivalent of prototype, provides explicit specification of all choice points. Only choice points can be configured, no direct external access to elements inside
	+ Providing a component and a property such as security?
	+ Name match and replace, in the environment configuration
	+ Array Sizes – Multiplicities reflected in Features, need to see that the port is an array. Internal subcomponent arrays mapped into feature array.
	+ Refined to – specify selection multiple levels down in architecture design – could be provided for V2.3
	+ Inheritance & Overriding of Property Values
* 1130-1230: AADL v3 discussions (array connections, unified type systems)
	+ **Action**: Brian – has a formula for expressing arrays, will send to Peter.
	+ Peter, want to expose the arrays at the high level so redundancy is understood.
	+ Provided an approach that fits with Brendan’s considerations for expression.
	+ Please try out what Peter has covered. **Action**: Peter to send out slides on arrays.
	+ Unifying the type system
		- Types can have properties
		- Unions and aggregates. Aggregates records, arrays, sets, multiset(bag), list (sequence), map, graph
		- Can use a type in the type system since its defined.
		- Property definitions reference types
		- May be able to simplify data annex.
		- Do we need implementations now, Peter believes not, sent examples, Denis sent one that Peter discussed.
		- Packages provide dot separated IDs as names, request to offer the same for property sets.
		- Property sets have simple ID as name
		- Properties can be in multiple stereo types is acceptable (for safety and for security as an example). Specification of which properties are relevant for given analysis.
		- Assists in formal relationships. Constraints.
		- MetaData – structural design constraints properties.
		- Reference types, meta model classes.
		- So we need some dividing line.
		- Need expressions, currently not part of property sublanguage, operators, built-in functions, user definable functions
		- Behavioral and temporal design constraints – should we use PSL
		- OCL and Alloy, Alloy is a user friendly form of OCL, well defined, documented.
		- Constraints annex – structural and behavior, expression language.
		- So should we start with OCL or Alloy for structural, PSL – send Peter your opinion.
		- Lutz will work on the Unified Type system, properties. Brian wants types and units with the metamodel, when would something be available.
		- Xtext, Xtypes, Xsemantics – looking at too.
* 1230-1400: Lunch
* 1400-1500: Continued (array connections, unified type systems)
* 1500-1530: Break
* 1530-1600: AADL v3 discussions (Configuration & Binding) (Peter Feiler, Alexey Khoroshilov, Denis Buzdalov)
	+ Bindings -
* 1600-1730: AADL Core and EMV2 Errata, semantics of XOR in EMV2 white paper (Peter Feiler)
	+ Input from Altarica possible
	+ Unspecified (a certain error type is not listed on the left) – Unspecified (also not listed on the right) -> completeness could only be with respect to the defined ontology for errors. Can subset the error types in the ontology to be more specific.
	+ Discussion – adding this product types, Brian has presented the slides in Oct. There is more that we need. Peter – not ready to consider this now. Later for value of product types (bigger).
	+ A1 xor A2 Xor A3 should be any of these should trigger
	+ Event and state based condition evaluation
	+ Error events are occurring independently
		- Only one at a time can trigger a transition
		- We want k of n operator – use case: triple redundant sensors
		- If one fails go into degraded mode
		- If 2 or more fail go into failstop model
		- Support arrays 1 ormore (p1, p2, p3)
		- Binding points: potentially unknown# of propagation sources
		- Proposal for next meeting (**Action:** Peter next meeting we will finalize, what if a port does not make any difference.)
	+ Binding points
		- Processor provides resources
		- Thread consumes resources
* 1730-1800: Tree of Flow specifications Recommendation (Steve Vestal)
	+ Tree of sensors into a system component.
	+ Typically would analyze each flow into the merge point.
	+ Take the one of interest – slowest.
	+ Could be related to time stamps
	+ How could we specify a merge point 2 inputs with one output or graph representation.
	+ Do you need a property to understand what you want at the merge point?
	+ Property identifies the merge point and what flows are included.
	+ 1123

# Wednesday, Feb 1

* 0900-1000: Demonstration of BA formal semantics Project (Jean-Pierre Talpin)
	+ Project for Airbus, 3 years, now completing.
	+ Signal, Polychrony, AADL, formal expression of behavior annex, step by step
	+ Micro steps definition of the automaton until a complete state is reached of the model invalidated.
	+ Supports synchronous and asynchronous behavior.
	+ AADL-BA to Signal
		- Modeling AADL timing semantics in Signal
		- Frozen time event on the input data ports
		- Thread\_behavior
		- Output data port
	+ SIGNAL thread
		- Each step in AADL would have micro-steps
		- Process with an implicit oversampling
		- Outputs are delayed compared with inputs
		- Has a Dispatch\_Thread input event
		- …
	+ Dynamic Proof, demonstrate that P never occurs. Export from SIGNAL
	+ Prove that the dispatch thread is a complete thread.
	+ Works for C code
	+ Supports hierarchical clocks, works with Lustre,
	+ **Action**: Jean-Pierre to send Brian the BNF for transistions, action – the encoding in SIGNAL.
	+ Work in Progress … see presentation.
	+ This is a demonstration but not yet a full fledged proof, starting to use model checkers.
	+ Proving that outFullBreak and outSpeedUp are exclusive
	+ Question: Concurrency protocol – how does it translate it – could
	+ Question - Relationship between two computers and their threads, relationship to the clocks, but not completely synchronous, can you express – could be doable
	+ Clock domains – expressing in AADL? Yes but you may need to add properties, there are clock domains and some properties but we would like to improve and open to input. Latency calculation in OSATE takes clock domains into account, synch and asynch.
	+ C
* 1000-1030: Merging BLESS and JP’s formal semantics for SCADE Expression (Brian Larson)
	+ SCADE AADL and BLESS Experiment
		- Gui Goretkin of ANSYS has assisted and very responsive
		- BLESS is BA with non-executed, temporal-logic, assertions. Logic is first-order predicate calculus augmented with two, simple, temporal operators, @ (continuous) , ^ (periodic). Defining when predicates are evaluated.
		- Assertions on ports, together with component invariant declaratively specify behavior.
		- BLESS has formal definitions of each BA construct. It’s a super-set.
		- Theorem Proving framework constructed and executed. Inductive proofs guided by human interaction.
		- If supported by DO-178c qualifiable code generation, then it will execute in deployment as proven.
		- Simulation – SCADE provides a simulation capability for BLESS or BA. Simulation of the BA, the assertion part is non executable. Translating BA into SCADE.
		- Provides table of comparison of what is supported. Does the BA have last? BLESS has a tick with the next clock cycle assertion which is beyond the BA. Only for periodic threads, only on left side. Local variable assumption, not for shared variable. No, not for shared, hard to provide correctness. BLESS does not forbid but difficult to prove. It’s outside my control. Could be in a relationship of assumes guarantee (I know). Ports are like a stream, not a local or shared variables. The semantic of a port is much more clean. Many errors in systems that do not control port (freeze). BLESS could work in the shifted back aspect as well.
		- Demonstration – Isolette, with big clock which is the same as the big step in JP example. It triggers dispatches.
		- About 10% through all aspects of what I would like to do.
		- Try Architecture + Suite state machines -> Bless verification.
		- Could use Polychrony to do timing analysis, BLESS for proof, export BA to SCADE for simulation. Working Scade to BLESS.

* 1030-1100: break
* 1100-1130: Merging BLESS continued
* 1200-1230: Discussion of next steps (Etienne Borde)
	+ Ballot to start next week
	+ Formalization we have been talking about for some time, we have a good idea and documents prepared with the formalization. How does it impact the way BA is used and carried forward?
	+ JP does not have much impact, it’s just on the state machines.
	+ It’s fitting with BLESS formalization although it’s a different approach.
	+ Differences in syntax and usage – the now operator. Turns into a new runtime service to get current time. Brian – that is very good. Next and previous – JP’s approach is computed from BA, BLESS should be able to compute from the BA as well. Brian – I agree. Not put the operators in BA and BLESS infers the Next and previous. JP has different kinds of constraints he wants to prove. Brian – would like to explore since BLESS might be able to support. Propose a Behavior constraints annex. That allows us the capabilities to maintain the two. You have two options, have a constraint language that expresses everything you might want to prove. You also have different expression languages. BA could have a sub-annex as an option.
	+ The first option is not really working – too complex and large. Second option would include sub-annex. We should have sub-annex. Would work together. Tools that do not supported would ignore it. Assertions are not the only reason we need sub-annexes. The idea of constraints separate have been from the beginning. What is the behavior and what is the requirements of the behavior
	+ Opening the core textual model, then see the annex annotations, like a graphical and see the available constraints. Smalltalk type textual windows. Desired features of OSATE. What we say in the standard is that its embedded, not stored separately. You can store the error annex in a separate file. I go to the type, then see the subareas.
	+ Need to work across annexes.
	+ How do we express across annexes and sub-annexes? Once we understand this, then we can build the sub-annex. We must separate the behavior annex and the annotations. Formal specification of the runtime service definitions would be very helpful (in core).
	+ Brain will attempt a formalization of the runtime services.
	+ Version 3 will be organized differently. Runtime services. Property language
	+ Syntax, semantics, formal semantics for runtime services. Would be very helpful for code generation annex. Etienne’s **Recommendation**: Before the formalization of semantics, let’s have a look at the code generation annex and its services, then the core runtime services to insure consistency. Then go to the formalization annotations as a separate annex.
	+
* 1230-1400: Lunch
* 1400-1430: Ellidiss Update on AADL tools, technology, and Projects (Pierre Dissaux)
	+ AADL Tools with Ellidiss
	+ 10 years + investment in AADL, New Technology
	+ 20 years + HOOD support
	+ Design: Stood for AADL, project management, instance model graphical editor, requirements traceability, documentation generator, Export textual AADL, Multi-user, Multi-project
	+ Verification: AADL Inspector
		- Import textual AADL, model processing plugins (static rules checkers, Scheduling analysis (Cheddar, Simulation (Marzhin), Pre-processors (import UML profiles (MARTE, SysML, …), Import Domain Specific models
		- …
	+ Top-Down modeling process for AADL
		- Provides design approach for AADL, insights from biggest European avionics projects.
		- Safe subcontracting (sub-trees), rigorous visibility rules
		- Ease testing, integration and maintenance
		- Detailed design (structured text)
	+ AADL Inspector – imports Declarative AADL model
		- Cheddar, Ocarina, Projects browser, Simulation (Marzhin), Simulation I/O, AADL model core + annexes, we keep Cheddar and Marzhin alined so consistent where they overlap. Code generation. LMP inside.
		- Multi-core, multi-processor SMP supported by Cheddar,
		- OCARINA, had to work with project environment, settings to integrate.
		- Now extended multi-project, AADL can be distributed then will be pulled together from repositories.
		- Scenario driven simulation
	+ Marzhin – multi-agent real-time simulator.
	+ No generic solution for importing SysML, UML, MARTE because of variants but we have one variant that people can build from.
	+ LMP makes all the connections of capabilities
	+ Cheddar - Multicore scheduling (SMP, BMP), cache aware scheduling, network on chip scheduling analysis, partitioning methods, multi objective optimization
	+ European Robotic Goal-Oriented Autonomous Controller – started last year, EU funded
		- Using the TASTE toolchain
		- Robotic engineers specific front end
	+ [www.h2020-ergo.eu](http://www.h2020-ergo.eu)
	+ Larger project Space Robotics: PERASPERA
	+ Safety Analysis support in AADL Inspector (interface with COMPASS but still some parts specialized)
	+ AADL subsets (working with Jerome) need to formally specify the part of AADL used by the analysis tool, we will support this even if it does not become a part of the standard.
	+ SCADE AADL editor overview
		- Prototype made from the Metamodel for AADL which includes semantics
		- Do not have modes yet
		- Graphical shapes not developed yet, but use colors to help distinguish
		- Allocation is separate from the component in SCADE
		- Can integrate the platform model with the software component model and then bind.
		- Merging changes between SCADE system and SCADE architect
	+ Only had data ports in the diagram – need to have event and event data ports. Security Modeling & Analysis – Telecom Bretagn/French Navy (PhD student)
		- Need to have a list of what is needed such as the event and event data. They should provide across the tools. Synchronizer.
		- SCADE architect is for software architecture.
		- Three different metamodels the avionics, FACE, and AADL plus aviation buses, AFDX, A653, A429, can, Loco\_Engine
* 1430-1500: Hybrid Annex discussion and Physical Modeling (Brian Larson, Pavi)
* 1500-1530: Break
* 1530-1600: Considerations for Cyber Physical Systems (Peter Feiler)
	+ Physical feature – have the need to talk about features that have continuous time
	+ Abstract feature – attach properties
	+ Observable feature – getting internal state without a connection.
	+ Need some examples
	+ Directional, non-directional, bidirectional
	+ Introduce a continuous port or have a property that is physical for an abstract feature
	+ We would need to collect the modeling needs, is it more like a bus, a pipe for information
	+ Transfer mechanism – has weight, length, type of flow.
	+ Physical – Can use generic concept of abstract, need perhaps a way to express the continuous variable via a hybrid annex or types that represent continuous variables. Device is our way to interface with the physical world. Attaching properties are a way that we can define the generic nature of an abstract. Physical type does not require a physical feature. Features do not have to be ports. Attach the property that you are dealing with. Move forward with a hybrid annex.
* 1630-1700: Planning for next meeting (Bruce Lewis)
	+ Notes for general topic of next meeting
		- **Action:** Alexey and Denis may want to provide updates/errata for the 653 Annex (30)
		- Definitions of AADL Runtime Services – Etienne would like to present (1)
		- Add Denis as a member before Ballot goes out.
	+ Do we have someone to take over the update of the A653 annex?
	+ SEI will likely have someone to take over security annex but working with Yves and others interested in contribution. Action: Lay out the scope of the security annex, need meeting Yves, Adventium, Dave Gluch, send out email to see who is interested in being part of the planning. Kansas State also. Adventium includes Steve and Charlie.
	+ Latest version of specification, can we have each a copy so we verify it across the committee.
	+ Copyright – Bruce will get contact, SEI has a capability to get permission.

1700-1730: Additional time for V3 if needed (Peter Feiler)

# Thursday, Feb 2

* 0900-0930: Assurance Cases – Integration with the Requirements Annex (Peter Feiler)
* 0930-1130: ALISA Tutorial (Peter Feiler)
	+ Architecture Lead Incremental System Assurance
		- Assurance: Sufficient evidence that a system implementation meets system requirements
	+ Part of our Assurance & Qualification Improvement Strategy
	+ Architecture –led Requirements Spec with ACVIP
	+ Requirements 50% user, 50% derived
	+ Incomplete and missing is more than 50% of requirements errors
	+ Incomplete 21, Missing 33, Incorrect 24, ambiguous 6, Inconsistent 5
	+ Managed awareness of requirement uncertainty reduces requirement changes by 50%
	+ 80% of requirements changes from development team
	+ Expert requirement uncertainty assessment – volatility, Impact, Precedence, …
	+ System interactions, state, behavior - State means persistent state, behavior is the transformation from input to output, controls and resources are external, input and output all part of the requirements coverage space,
	+ Design and operational quality attributes like Performance, Modifiability, Availability, Security etc. are also part of the requirements space.
	+ Fault impact & contributors – Hazards also part of the Requirements space.
	+ Three dimensions of Incremental Assurance
		- Incremental assurance through virtual system integration
		- Priority focused architecture design exploration,
		- Compositional verification and partitions to limit assurance impact.
	+ Modeling Notations in ALISA Prototype
		- Represent goals, systems requirements, hazards
		- Specify intended verification activities
		- Specify compositional verification and assurance plans
		- Manage assurance state and results
	+ Will be by project phase and domain (like safety)
	+ Requirements Definition and Assurance for AADL – RDAL
	+ Reqspec is a textual notation to capture the specification including goal oriented architecture led requirements, requirements decomposition, integration with safety analysis via Error Model Annex V2, bridge to existing requirements documentation practices. Stakeholder Goals – may be in conflict, System Requirement Specification – consistent set of system requirements that must be met.
	+ Stakeholder Goals Notation
		- Can be associated with AADL model element if desired
		- Description, rationale
		- Refinement, evolution, conflict resolution
		- Traceability to stakeholders and DOORS requirement
	+ Use Scenarios
		- Leave external – reference document
		- Incorporate – capture in the ReqSpec
	+ System Requirement Specification Notation
		- Verifiable contract via verification plans – satisfiable design goal
		- Associated with AADL model element
		- Description, rationale
		- Parameterization by variable
		- Refinement, decomposition, inheritance, evolution
		- Verifiable by verification activities
		- Formal specification of predicate
		- Traceability to goals, hazards
		- Traceability to DOORS and other documents.
		- Refinement is to make it more precise, decomposition is what parts affected. If you attach it to a subsystem, it’s a decomposition, if clarifying the system requirement, it refines.
	+ Clear system boundary is critical.
	+ Report generator has been hooked to ALISA
	+ Export/Import through ReqIf. Interchange tooling and meta model for requirements.
	+ Requirements get changed a lot, often those with values. So this constaint can be in many places. Define for requirements set or individual requirements.
	+ Consistency between val and AADL properties Maxweight=1.2kg, then can be passed in to verification method.
	+ Val can be used to compute to see if the assurance case is satisfied.
	+ Global requirements apply to the whole model. “All features of a component are connected.”
	+ “and” “or” logic is not sufficient – and stops evaluation at the first failure.
	+ Multi-valued Assurance Logic in Verification Plan
	+ Registry of Verification Methods – can call existing OSATE Analysis plugins, or Resolute claim functions, or reflective execution of Java/Xtend functions, Execution Junit tests, more to come (Agree, Simulink, SCADE, manual.
	+ You can write your own verification methods that take the place of a plug-in. Simple interface.
	+ Verification Plan – what methods you run at what point.

# Webex Info for the meeting:

AS-2C AADL Committee Meeting

Every day, from Monday, January 30, 2017, to Thursday, February 2, 2017

8:30 am | Europe Time (Paris, GMT+01:00) | 10 hrs

JOIN WEBEX MEETING

<https://sae.webex.com/sae/j.php?MTID=mf02b56d11ab0cef984998605b8933c8d>

Meeting number: 629 078 360

Password: AS2Caadl