Introduction to AADL analysis and modeling with FACE Units of Conformance

Distribution Statement A: Approved for public release; distribution unlimited.
AMRDEC Aviation Applied Technology Directorate Contract Number W911W6-17-D-0003 Delivery Order 3

This material is based upon work supported by the U.S. Army Research Development and Engineering Command (RDECOM), Aviation & Missile Research Engineering, Development and Engineering Center (AMRDEC), Aviation Development Directorate (ADD) under contract no. W911W6-17-D-0003. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Army.

Revision: August 27, 2018

Why AADL?

The Architecture and Analysis Design Language (AADL) is a Society of Automotive Engineers (SAE) aerospace standard (AS) system model specification language (AS5506C) that supports various types of performance and safety analysis. The Future Airborne Capability Environment (FACE) Technical Standard defines a Reference Architecture intended for the development of portable software components targeted for general purpose, safety, and/or security purposes. The FACE and AADL standards are complementary. AADL provides tools for modeling and analyzing aspects of integrated cyber-physical systems that are outside of the scope of the FACE Technical Standard. AADL adds computer hardware and software representations and hierarchical computing system compositions to the rich software data structures captured in the FACE Technical Standard, providing additional capabilities for detection and prevention of hardware implementation errors. Using the FACE Technical Standard and AADL together provides for early detection of many classes of integration errors.

What is AADL?

Per SAE AS5506C, AADL is a semantically precise modeling language for describing both the software architecture and the execution platform architectures of performance-critical, embedded, real-time systems using standardized textual and graphical representations. AADL focuses on model-based analysis of static and dynamic computing system properties. AADL supports modeling at various levels of detail through compositional computing system specification and model refinements and extensions. For example, a high-level system model might have an abstract Transport Service Segment (TSS) library that is refined in subsequent models to a Common Object Record Broker Architecture (CORBA)-based TSS library or to a Transport Control Protocol/Internet Protocol (TCP/IP)-based TSS library. There are analyses for many performance and quality metrics and there are tools to help integrate systems. For example, AADL analyses allow you to evaluate the costs and benefits to size weight and power (SWaP) and performance of different refinements.

How does AADL relate to the FACE Technical Standard?

The FACE Technical Standard provides a means to describe Units of Conformance (UoCs) or Units of Portability (UoP) and the data they exchange, whereas AADL provides the capability to model a cyber-physical system constructed from FACE UoCs and extended with hardware and operational properties. AADL supports modeling software structures such as data types and threads, hardware bindings,

and environmental context to facilitate analysis of the interactions between system components in both the hardware and software domains. AADL and the FACE Technical Standard overlap in their capacity to describe some system features but focus on different system characteristics and on highlighting different classes of errors (see Figure 1). AADL is particularly good at detecting errors that only manifest in integrated computing systems. For example, AADL analyses can detect conditions like total memory exceeded, fully loaded processors fail to meet all deadlines, or unhandled error propagation flows.

How do you start using AADL?

A variety of tools support AADL. A commonly used tool is the Open Source AADL Tool Environment (OSATE), which is an Eclipse-based editor that comes with analysis tools. OSATE and the accompanying tools are maintained by the Software Engineering Institute (SEI) at Carnegie-Mellon. The free download of the most current version of OSATE can be found at http://osate.org/download-and-install.html. Once the chosen version of OSATE is downloaded, unzip the archive and run "osate.exe" to open OSATE.

AADL models are organized into projects. A new project is started by going to File>New>AADL Project. Non-AADL files, such as .face files, can also reside in AADL projects. A system, when modeled in AADL, can be separated into different models and packages that reference each other.

How do you work with a FACE Data Model in AADL?

The AADL Annex for the FACE Technical Standard describes the specific mapping from FACE Data Models to AADL. The SEI is in the process of developing a FACE Data Model to AADL translator, which creates an AADL model derived from a given FACE Data Model.

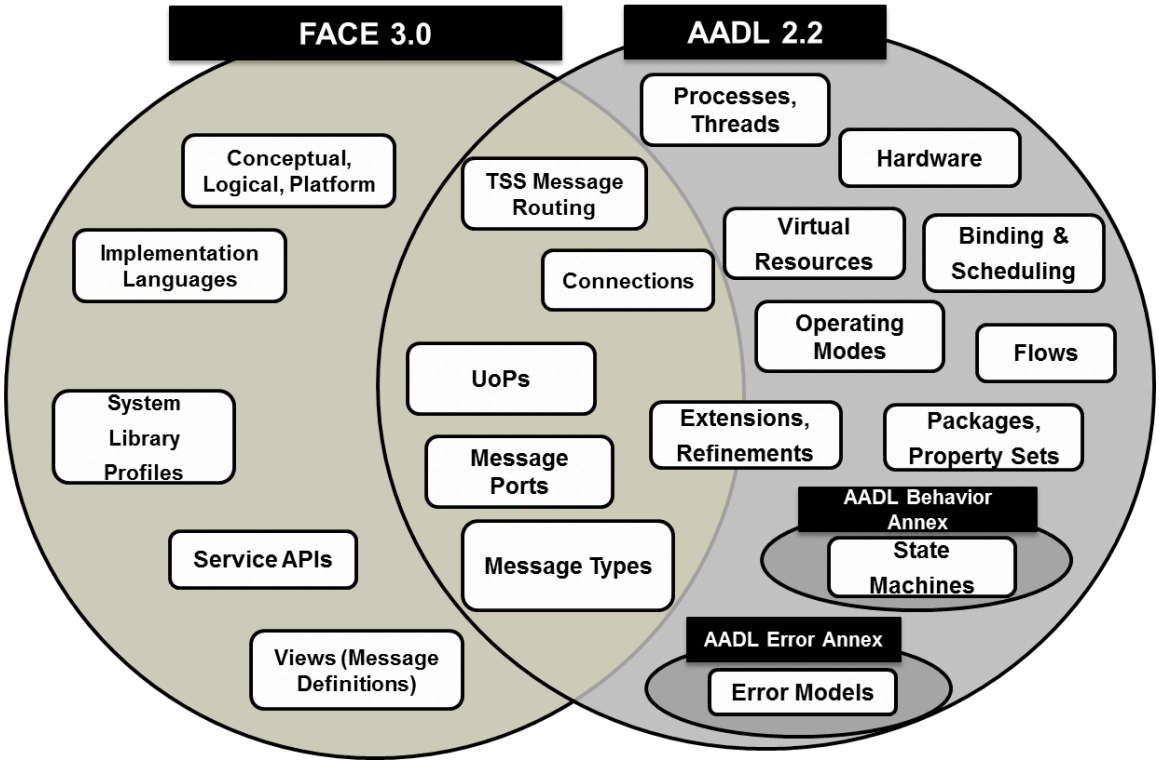

Figure 1 Overlap between AADL and the FACE Technical Standard

The entire FACE Data Model is not translated as there is not a 1-to-1 mapping of FACE to AADL artifacts, but the information in the FACE Data Model that is relevant for AADL analysis is translated to its equivalent representation in AADL.

To install the translator, download the plugin zip file available via OSATE web page. In OSATE go to Help>Install Additional OSATE Components and select the FACE Data Model to AADL Translator. Run the translator by right clicking on any .face file in OSATE and selecting the menu option "translate to AADL." The translator generates a folder named "model-gen" that contains the AADL translation of the .face file. Add further information such as runtime properties and hardware bindings to the generated model in AADL to support the desired analysis.

Right click on a system or component and select the menu option "Create Diagram." to create a visual representation of the generated model using the AADL Graphical Editor (see). The Graphical Editor generates user-editable diagrams of AADL component implementations.

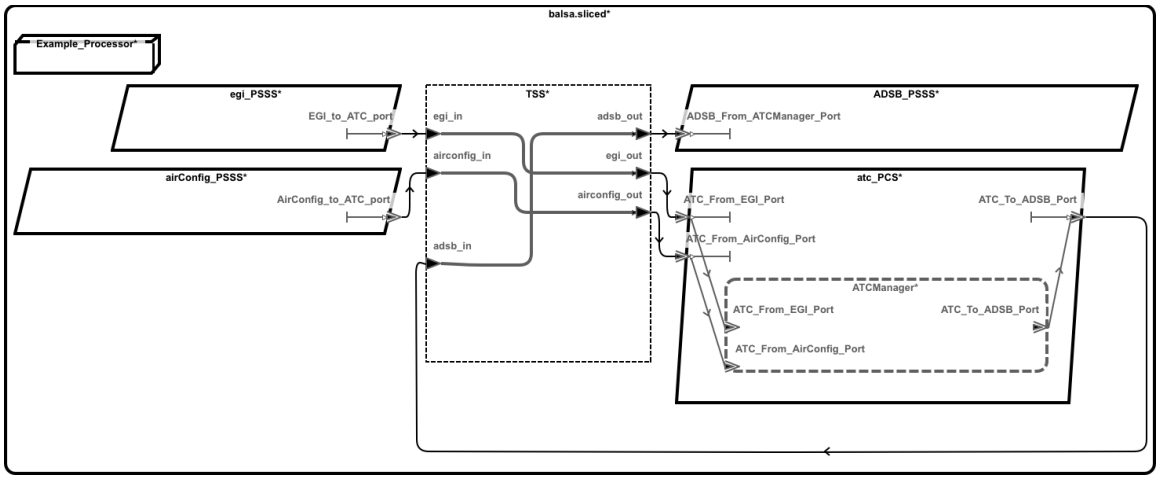

Figure 2 BALSA Modeled in AADL and shown in the AADL Graphical Editor

How do you use AADL to perform analysis?

A component modeled in AADL has three representations; a declaration defining its external features and interfaces, an implementation that defines its internal characteristics, and an instance that flattens the declaration and implementation into an analyzable format. The component declarations and implementations are written in the AADL textual model and the instance is created when the top-level system implementation containing the component is instantiated. Instantiate a system implementation by right clicking on it in the AADL Outline menu and selecting the menu option "instantiate." AADL analysis tools will parse instance models and recognize specific properties. For example, the standard OSATE plug-in "Check Flow Latency" uses data flow declarations as well as timing property tags to calculate predicted system latencies and compare them to declared limits. Analysis tools are invoked in OSATE by selecting an instance model and selecting the menu option for the desired tool. Additional analysis tools can be added to OSATE via plug-ins.

How do you model BALSA in AADL?

The Basic ADS-B Lightweight Source Archetype (BALSA) exemplar model is a working software example of applications aligned to the FACE Technical Standard executing in a FACE Reference Architecture. The BALSA model can be translated into AADL using the translator as mentioned above. The autogenerated AADL model for BALSA is populated with properties inferred or read directly from the FACE Data Model, such as periods for each of the threads. Additional properties, hardware bindings, and data flow information can be added to the BALSA model either through direct AADL model modification or by creating an extension of the model components which can be modified without changing the autogenerated model.

What resources are available for AADL?

Introductory AADL Materials

- AADL Video Tutorial 0: Introduction to AADL and OSATE https://www.youtube.com/watch?v=MjUQZaqaTA4
- AADL Tutorial 1: Components in AADL https://www.youtube.com/watch?v=vm_2FdPEFjk
- AADL Tutorial 2: Ports and Connections in AADL https://www.youtube.com/watch?v=c-xsWYxSCHY
- AADL Tutorial 3: Processor and Memory in AADL https://www.youtube.com/watch?v=HemoCKFmBjw
- AADL Tutorial 4: AADL Threads https://www.youtube.com/watch?v=E_5kJZe0u7c
- SEI Webinar Architecture Analysis with AADL (2014) https://www.sei.cmu.edu/webinars/view_webinar.cfm?webinarid=424907
- AADL In Practice: Become an expert of software architecture modeling and analysis by Julien Delange https:// www.amazon.com/AADL-Practice-software-architecture-modeling/dp/0692899642
- Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis and Design Language, 1st Edition, by Peter H. Feiler and David P. Gluch https://www.amazon.com/Model-Based-Engineering-AADL- Introduction-Architecture/dp/0134208897
- SEI AADL in Practice Workshop https://www.sei.cmu.edu/training/P128.cfm
- Georgia Tech open course on systems engineering that features AADL https://www.udacity.com/course/cyber-physical-systems-design-analysis--ud9876

AADL Tools

- OSATE http://osate.org/index.html
- Adventium CAMET Library (resource for AADL analysis tool plug ins, example models, and training) https://www.adventiumlabs.com/our-work/products-services/model-based-engineering-mbe-tools