ADEPT 2022 workshop: a summary of strengths and weaknesses of the AADL ecosystem

Frank Singhoff[1], Jérôme Hugues[2], Hai Nam Tran[1], Gianluca Bardaro[3], Dominique Blouin[4], Marco Bozzano[5], Patrick Denzler[6], Pierre Dissaux[7], Eric Senn[8], Xiong Xu[9], Zhibin Yang[10]

1. University of Brest, Lab-STICC UMR CNRS 6285, Brest, France, firstname.lastname@univ-brest.fr

2. Software Engineering Institute, Carnegie Mellon University, USA, jhugues@andrew.cmu.edu

3. Politecnico di Milano, Piazza Leonardo Da Vinci 32, Milano (IT); email: {name.surname}@polimi.it

4. LTCI, Telecom Paris, Institut Polytechnique de Paris, Palaiseau, France, dominique.blouin@telecom-paris.fr

5. Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy, bozzano@fbk.eu

6. Institute of Computer Engineering, TU Wien, Vienna; patrick.denzler@tuwien.ac.at

7. Ellidiss Technologies, 24 quai de la douane, 29200 Brest, France; pierre.dissaux@ellidiss.com.

8. Lab-STICC, Université de Bretagne Sud, Lorient, France; eric.senn@univ-ubs.fr

9. SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing, China; University of Chinese Academy of Sciences, Beijing, China; xux@ios.ac.cn

10. School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China; yangzhibin168@163.com

Abstract

The Architecture Analysis and Design Language (AADL) is a SAE Standard for the modeling of both the hardware and the software of embedded systems. The AADL standard is now mature and is today employed by numerous stakeholders in the domain of critical embedded real-time systems to address a large set of concerns: performances (latency, schedulability), safety, or security, ... The ADEPT workshop aims to present and report on current projects in the field of design, implementation, and verification of critical systems where AADL is a first-citizen technology. This article is a summary of the ADEPT 2022 workshop.

Keywords: AADL, critical embedded real-time systems, design, implementation and verification.

1 Introduction

The Architecture Analysis and Design Language (AADL) is a SAE standards for the modeling of both the hardware and the software of embedded systems [1]. The AADL standard is now a mature standard for the modeling of critical embedded real-time systems. AADL is today employed by numerous stakeholders in the domain of critical embedded real-time systems to address a large set of concerns: safety [13], security [15], or performance (latency, schedulability) [14] but also code generation [12, 21]. One key strength of AADL as a language is the set of tools that provide those analysis capabilities.

The ADEPT workshop aims to present and report on current projects in the field of design, implementation and verification of critical systems where AADL is a first citizen technology. The ADEPT workshop is also an opportunity for AADL beginners to meet experienced AADL practitioners.

The ADEPT 2022 workshop was a full day workshop. A morning session was introducing new tools and was an opportunity for AADL beginners to discover the language, its tools, and its potential uses. The afternoon was dedicated to the presentation of success stories and returns of experience in the form of a discussion with the workshop attendees. The workshop was co-located with the 26th Ada-Europe International Conference on Reliable Software Technologies (AEiC 2022) at Ghent, Belgium.

In the sequel, we describe the 2022 workshop program in section 2. Section 3 presents a summary of the return of experience reported by the workshop participants. Then we conclude is section 4.

2 Workshop program

The workshop gathered 22 participants, with 8 presentations. It was organized in 4 sessions: 1) an introduction from the workshop organizers about the AADL standard, its ecosystem and the ongoing standardization activities, 2) a session introducing emerging AADL tools, 3) a session presenting success stories, case studies, and return of experience, and 4) a discussion between the workshop participants leading to a review of the current strengths and weaknesses of the AADL standards and tools.

The tool session was composed of 4 presentations. [2] addresses SysML v2 and AADL. The speakers show how SysML v2 constructs can be mapped to AADL entities, allowing verifications on SySML models by AADL tools such as AADLInspector. The 3 next presentations were examples of how formal methods can be combined with AADL [3, 4, 5]. [3] introduces MARS, a graphical modeling and verification tool. MARS inputs are models combining AADL and Simulink/Stateflow components. Both AADL and Simulink components are translated to hybrid communicating sequential processes for formal verification purposes. [4] describes C2AADL_Reverse, which is both an approach and a tool for model-driven reverse engineering. The proposal is to extract design artefacts from a multi-tasked C code in order to automatically produce AADL models. The produced model can then be used for formal verification or code generation. Finally, presentation [5] was dealing with the COMPASTA project in which the COMPASS [11] and the TASTE [10] tools are combined. Both are AADL oriented tools: TASTE is a set of tools bridged by AADL models while COMPASS provides formal safety analysis on models expressed with a dialect of AADL.

The case study session hosted 4 presentations [6, 7, 8, 9]. Two presentations were addressing robotic systems modeling and verification [7, 8]. They are both focussing on the ROS platform and show how AADL is able to model ROS nodes, network and device entities. In [7], AADL modeling of ROS entities targets early performance verification to predict required network bandwidth, end to end latencies or processor utilization. [8] proposes to generate a part of the C/C++ code for ROS architecture from similar AADL models. Network modelling is also a concern in the talk [6] where the AADL is used to model a gateway architecture in the context of fog computing. One of the motivations of this AADL model is to generate complex gateway configurations. Finally, the last talk illustrates the use of AADL in the context of a railways system with ALISA (Architecture-Led Incremental System Assurance) [9]. In this context, ALISA is used to both express and verify system requirements of a train control system.

3 Strengths and weaknesses of the AADL ecosystem

During the workshop, a session dedicated to a discussion about return of experience from workshop participants has been organized. Return of experience collected during this session is summarised in the sequel. 3 topics were discussed: topics related to the AADL language, its tools and its community.

About the AADL language, the main addressed topics covered the rationale to choose AADL. The participants also identified the language features that are missing today in AADL and the ones they wish for in the next versions of the standards.

Availability of tools is one of the motivations to select a design or a programming language. AADL tools used by the workshop participants have been enumerated during the workshop. Participants also reviewed the main issues they faced when using them. Such review is an opportunity to understand the future features workshop participants are waiting for in the next tool versions.

Finally, model based technologies such as AADL technologies strongly change how practitioners design and implement software. Having an active user community contributes to increasing stakeholders AADL skills and then contributes to successfully applying AADL on concrete projects. What AADL users expect from the user community and how this community would be organized were also topics discussed during the workshop.

In the sequel, we only describe the items reported by workshop participants that we believe are more important.

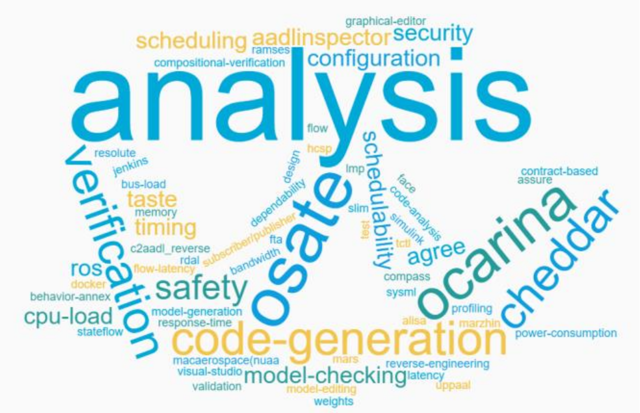

Figure 1: User's intends and used tools

3.1 Rationale to choose AADL

Workshop participants explained why they have chosen AADL for the work they presented. Sometimes, AADL was simply chosen because it was part of a larger toolset (e.g. TASTE) they are using. Sometimes also, as with any technologies, AADL was chosen simply because participants knew it before.

About the language itself, AADL was frequently selected because it has the ability to both model hardware and software components, with both textual and graphical syntaxes, and with features that bring software engineering good practices (e.g. inheritance, abstract components, modularity, …).

AADL's ability to model both static and dynamic aspects of embedded systems architecture and to predict its behavior is also an important rationale for most of the workshop participants. Participants aim to assess properties on the architecture behavior or to do design space exploration by investigating various architecture alternatives (e.g.

deployment options), to identify, for example, performance bottlenecks or safety/security vulnerabilities.

Last but least, the availability of AADL mature tools was an argument for several workshop participants. The list of AADL tools workshop participants are using is displayed in the word cloud of Figure 1. This figure also contains words representing the concerns of the workshop participants when using AADL. Not surprisingly, and it is easy to see it in Figure 1, participants expect analysis features on various aspects while simultaneously keeping mandatory features such as modeling and code generation services/tools. Finally, many tools exist for AADL, but one aspect which was highlighted by workshop participants is the ability of AADL to build heterogeneous tool-chains (e.g., TASTE, OSATE, AADLInspector).

3.2 AADL standard review

In the sequel, we enumerate several issues in the AADL standards which were pointed out by workshop participants. One of the workshop objectives was to identify the features that are currently missing in the AADL standards.

First, we must notice that several participants underlined the (too) large number of features already defined in the AADL standards. One of the expressed needs by workshop participants is to instead provide a better formal specification of the current AADL features and to define and support well-defined subsets of AADL standards. Some concepts are also expected to be simplified (e.g. flow specification is seen as cumbersome). Furthermore, workshop participants also missed proper documentation of standard constructs and options, to reduce the learning curve of AADL for beginners. We see there that improving the current AADL standard but with the same feature bounding box would be a first user requirement.

There are however few missing features that were identified by workshop participants. One of them is about safety and behavioral specification (e.g. specification of timed/hybrid behavior in state machines, bridge between behavioral and safety annexe). Workshop participants also remind that the standard proposes nothing today to model the physical environment of the system to implement.

Finally, AADL has extensions mechanisms that can be solutions for some of the requests discussed during the workshop, but the spirit of those mechanisms have probably to be more explained to practitioners. For example, workshop participants complain about the lack for the modelling of multi-core systems, machine learning components, ROS components, specific aspects of virtual bus or resource binding, while the current AADL standard provides extension mechanisms to allow users to define their own domain or application specific properties and property sets. Assessing that those extensions are able to cope with the requirements expressed by workshop participants stays an open question.

3.3 AADL tools issues and expected features

During the workshop, participants have also made a return of experience on the AADL tools they are using. One of the main reported issues is the level of compliance to the standard. Most of the tools cover only a part of the standard and as it is difficult to extend them by users, such limits restrict tool applicability. Tool interoperability would be probably better experimented by tool developers.

An important identified lack is about the relationships between AADL declarative models and instance models. There is a need of bi-directional transformation/flow between AADL declarative and instance models. Most of the tools only provide transformations from declarative models to instance models, but few provide the reverse after analysis for example. Second, an interest for transformation from or to a graphical representation is also pointed out.

Of course, AADL tools, as any software, are subject to bugs, deprecated features or release, poor documentation, unmaintained tools, and it was also pointed out by workshop participants.

3.4 AADL community

The last topic discussed during the workshop dealt with the AADL community and what users expect from it.

First, several participants highlighted that it is difficult to reach other AADL users. There is a demand for Internet forums animated to share experience, material, problems and solutions. Workshop participants expressed the needs of workshops, open source teaching materials, case studies, or working examples, in fact any materials that contribute to reducing learning time for AADL newcomers.

Few repositories and websites providing a part of such AADL material exist however: model repositories and examples with OpenAADL and AADLib [16], AADL cook books [17], AADL labs or teaching materials [18, 19, 20], … Those repositories and websites probably have to be better advertised but also updated to cope with user’s expectations.

Workshop participants also raised the interest to continue the AADL users' days that were taking place during the AADL standardization committee meetings. It shows that there is also a need for meetings devoted to users where open source tools/materials can be presented, demonstrated. An annual workshop as ADEPT could play that role.

4 Conclusion

AADL is a set of SAE international standards that aims to improve the quality of the critical embedded systems design. The AADL standards are now mature. The objective of the ADEPT workshop was to encourage discussion between members of the AADL community. ADEPT may be a location to share experiences on AADL and its ecosystem. In this article, we summarized the discussions during the first edition of the ADEPT workshop co-located with the 26th Ada-Europe International Conference on Reliable Software Technologies (AEiC 2022) at Ghent, Belgium.

Obviously, we cannot state as representative the return of experience presented in this article as it is the opinion of 22 workshop participants only. However, the workshop shows that AADL is an active standard currently used in various domains in Europe. The workshop pointed out also that the community lacks events to share experiences on the standards and its tools.

Acknowledgments

We would like to thank Ellidiss Technologies who supported the registration fees for the speakers of the workshop.

References

[1] *Architecture Analysis and Design Language (AADL) AS-5506c:* Tech. Rep. The Engineering Society For Advancing Mobility Land Sea Air and Space, Aerospace Information Report (2017), Version 2.2.

[2] Jean-Charles Roger, Pierre Dissaux. *AADL modelling with SysML v2.* Ada User journal, volume 44, number 1, March 2023, pages 63-66. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[3] Xiong Xu, Shuling Wang, Bohua Zhan, Xiangyu Jin, Naijun Zhan, Jean-Pierre Talpin. *Unified graphical co-modeling, analysis and verification of cyber-physical systems by combining AADL and Simulink/Stateflow.* Ada User journal, volume 44, number 1, March 2023, pages 67-70. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[4] Zhibin Yang, Zhikai Qiu, Yong Zhou, Zhiqiu Huang, Jean-Paul Bodeveix, Mamoun Filali. *C2AADL_Reverse: A model-driven reverse engineering approach for development and verification of safety-critical software.* Ada User journal, volume 44, number 1, March 2023, pages 71-74. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[5] Alberto Bombardelli, Marco Bozzano, Roberto Cavada, Alessandro Cimatti, Alberto Griggio, Massimo Nazaria, Edoardo Nicolodi, Stefano Tonetta, Gianni Zampedri. *COMPASTA: Integrating COMPASS Functionality into TASTE.* Ada User journal, volume 44, number 1, March 2023, pages 75-78. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[6] Patrick Denzler, Daniel Ramsauer, Daniel Scheuchenstuhl, Wolfgang Kastner. *Experiences Modeling a OPC UA / DDS Gateway in AADL in the Context of Fog Computing.* Ada User journal, volume 44, number 1, March 2023, pages 79-79. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[7] Gianluca Bardaro, Matteo Matteucci. *Modelling robot architectures with AADL.* Ada User journal, volume 44, number 1, March 2023, pages 80-83. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[8] Eric Senn, Lucie Bourdon. *Modeling ROS based applications with AADL.* Ada User journal, volume 44, number 1, March 2023, pages 84-87. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[9] Dominique Blouin, Paolo Crisafulli, Françoise Caron, Cristian Maxime. *An Introduction to ALISA and an Experience Report on its Usage for an Industrial Railway System Case Study.* Ada User journal, volume 44, number 1, March 2023, pages 88-91. Also presented in the ADEPT 2022 workshop, Ghent, Belgium, 2022.

[10] Perrotin, Maxime, et al. *TASTE: An open-source tool-chain for embedded system and software development.* Embedded Real Time Software and Systems (ERTS2012). 2012, France, Toulouse.

[11] Bozzano, M., Cavada, R., Cimatti, A., Katoen, J. P., Nguyen, V., Noll, T., & Olive, X. *Formal verification and validation of AADL models.* In ERTS2 2010, Embedded Real Time Software & Systems 2010, May, France, Toulouse.

[12] Hugues, J., Zalila, B., Pautet, L., & Kordon, F. *From the prototype to the final embedded system using the Ocarina AADL tool suite.* ACM Transactions on Embedded Computing Systems (TECS), 7(4), 1-25, 2008.

[13] Delange, J., & Feiler, P. *Architecture fault modeling with the AADL error-model annex.* In 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications (pp. 361-368). IEEE, 2014.

[14] Singhoff, F., Legrand, J., Nana, L., & Marcé. *Scheduling and memory requirements analysis with AADL.* ACM SIGAda Ada Letters, 25(4), 1-10, 2005.

[15] Ellison, R., Hudak, J., Kazman, R., Woody, C., & Householder, A. *Extending aadl for security design assurance of cyber physical systems.* CMU/SEI, USA, 2015.

[16] AADLib repositories, http://www.openaadl.org/

[17] AADL cook book, http://beru.univ-brest.fr/AACB/

[18] Cyber-Physical Systems Design & Analysis course teaching, by Georgia Institute of Technology, https://www.udacity.com/course/cyber-physical-systems-design-analysis--ud9876

[19] CPS-IoT Summer School 2022, https://mem4csd.telecom-paristech.fr/blog/index.php/training-schools/cps-iot-summer-school-2022/

[20] SEA UE Teaching week 2022, http://beru.univ-brest.fr/cheddar/contribs/educational/ubo/SEA-UE_TEACHING_WEEK_2022/LAB/tp.html

[21] F. Cadoret, E. Borde, S. Gardoll and L. Pautet, "Design Patterns for Rule-Based Refinement of Safety Critical Embedded Systems Models," 2012 IEEE 17th International Conference on Engineering of Complex Computer Systems, Paris, France, 2012, pp. 67-76, doi: 10.1109/ICECCS20050.2012.6299202.