The Latest Developments in AADL

featuring Julien Delange & Peter Feiler interviewed by Bill Pollak

---

Bill Pollak: Welcome to the SEI podcast series, a production of the Carnegie Mellon Software Engineering Institute. The SEI is a federally funded research and development center at Carnegie Mellon University in Pittsburgh, Pennsylvania. A transcript of today's podcast is posted on the SEI website at sei.cmu.edu/podcasts. My name is Bill Pollak, and today I am pleased to introduce to you Peter Feiler and Julien Delange, both senior researchers at the SEI. In today's podcast, we will be discussing the latest developments with the Architecture Analysis and Design Language (AADL) standard. Before we dive into that topic, a few words about today's guests.

Peter is the technical lead and author of the Architecture Analysis and Design Language standard, which was published in November 2004. Version 2 of the standard was published in January 2009. He is a senior member of the technical staff. His research interests include dependable real time systems, architecture languages for embedded systems, and predictable system analysis and engineering.

Julien is a member of the technical staff at the SEI and leads several projects related to architecture modeling and avionics system design. His work on the AADL committee includes authoring the ARINC 653 annex for the modeling of avionics architecture. He also designed tools for the analysis and implementation of avionics architecture from models. He was also a lead designer and developer of the TASTE tool set, a model engineering framework for designing cyber-physical systems from models and supported by the European Space Agency. Welcome, Peter and Julien. Let's start off by having you tell us a little bit about the Architecture Analysis and Design Language standard and its role in software-intensive systems. How is AADL being used today?

Peter Feiler: Well, AADL aims at software-intensive systems as you say, and those are systems in which the software makes the system work. One first class of those systems is safety-critical systems like aircraft. And it's that industry that encountered that problem first. And they were looking for a solution, and it's in that context that we designed the Architecture Analysis & Design Language, AADL. The idea behind AADL is that we can discover problems early in the process, because industry data shows that 80 percent of software-related errors are discovered very late in the integration process. By that time, the rework cost is very high, and they have experienced cost explosions such that aircraft, 70 percent of the aircraft cost is now software.

Bill: So Peter, you were the technical lead and the author of AADL. Can you tell us a little bit about your role in developing it?

Peter: Well, it started out by looking over research in architectural languages in the 1990's. In particular, one language was developed by Steve Vestal at Honeywell called MetaH, specifically for embedded systems. What that meant was that it dealt with not only the software but the hardware that it runs on and the systems that it tries to interface with. And it's those three elements that are important for embedded systems and safety-critical systems, because it is the interface between them and the mismatched assumptions between those that usually cause these system-level problems that are caught late. Out of that, we started in 2000 with the standards effort, published a standard first in 2004. Industries started using it in pilot projects and as a result of that, we revised it and re-published a new version in 2009, with a small revision this fall.

My role in it was to bring technology from that one language together with ideas from other architecture languages and then, on the other hand, make it a practical solution based on the requirements of the industry. And the committee consisted of mostly industry folks who are trying to use the technology rather than researchers who are trying to push technology. And so my job was to bring it all together, make it a consistent language, and work it through the committee, so that it comes out the door and is ready for use.

Bill: Great. So I know that you work with critical software. Could you please relate some stories related to software errors from your experiences in the field, and tell us what role AADL might play in mitigating these errors?

Peter: One little story is a year ago I read Consumer Reports, and a washing machine failed its test. In the next issue, they had an article that the company sent a software patch. And after the software patch was installed on the washing machine, it passed the test. So it doesn't have to be a complicated system like a car or an aircraft that these kinds of problems happen [to].

At the same time, I got an article about the Ford Fusion hybrid car that had fires in its engine. It is caused by a software error they have recently identified. And there are numerous occasions

where aircraft are behaving erratically, or aircraft go into a nosedive without explanation. In many cases, what it is is that a fault, for example, in the hardware where two boards get too close to each other, and because of vibration, touch. It causes a transient error that corrupts data. And so when you get airspeed from this device, and it has this transient error, your aircraft thinks it's somewhere else and goes into a nosedive type of behavior. So what you have is when people do the system-safety analysis, they are focused on the physical parts failing, and they understand that part. But the consequence of that in software today is still not very well understood. And that's part of what we're promoting with AADL: not only just understanding the architecture but also understanding what's going on in the behavior of the system and especially then in the fault behavior of the system.

Julien Delange: And this is especially really difficult because you have more and more function relating to software. So you can have a trade-off, an issue when you have different function in the same processor and one function can then back another. So you have to be careful when you design your system, and to validate the system and to find issues before starting to develop the software.

The nice aspect of AADL is that you can perform some validation before starting the implementation. And also the semantics of the language is appropriate to automatically generate the code. By automatically generating the code, you make sure that the system's specifications are enforced in the implementation, and you avoid all the program traditional errors you have when you produce code manually with a human developer. Also you make sure that the interface between each function is enforced. One step later will be to also automate the tests of the system once you produce implementation, but this is an ongoing research topic, I guess.

Bill: So what do you see as the future of the Architecture Analysis and Design Language?

Peter: I guess there are three parts to an answer: the one part, what happens to the standard itself? The second part is, what happens to technology that goes around the standard and then, in its actual use? On the standard itself, the AADL standard is actually a standard suite. There is the base language, and then there are extensions that help you deal with certain issues.

One area in which we just are finishing the revision is actually the safety area, where we have what we call the error model annex. It allows you to attach fault behavior descriptions to your base architecture description, and in such a way that it supports you throughout your whole safety analysis process, starting out with functional hazard assessment through failure mode and effects analysis, fault-tree analysis to then actually designing your fault management architecture. So what we are doing—and that's a combination of what we are doing as a standard and also in the research community—is adding this error model extension.

We are also adding an extension that deals with formally capturing requirements and tracking how the requirements are supported. It's in that context we are moving in the direction of assurance cases, where we deal with the evidence that we are collecting through analysis together with testing and making sure that the whole thing is consistent. At the end, the idea is that we can do end-to-end verification and validation throughout the whole development process like Julien was already alluding to. In that context then improve the way we are doing not just testing, but also cut down the cost of certification and qualification of systems, and potentially do that incrementally.

Julien: Actually what the last researcher did was to validate the system before implementation. Right now we try to use a model to satisfy and automate the certification of the system. For example, you check that the behavior at runtime is consistent with the specification, which is very difficult to do without testing manually the system. So the idea also is to use AADL in its own domain. So right now for the safety analogies, we are applying this to the medical domain also. And this is also part of an effort to design a safe medical device.

Another research interest is to connect the AADL technology with other technology being developed at the SEI. For example, we are making some cost assessment of different iteration of the system architecture. So when you have a defined and established software architecture, what is the cost when you make a new iteration? When you change something in the architecture, what would be the cost and the impact? And you are also trained to connect AADL with that because AADL describes the run time aspect of the system's deployment, the configuration, and you can add some variable information that can have an impact on the costs.

Peter: Just building on the medical device domain, it is actually the FDA [the United States Food & Drug Administration] that is quite interested in that kind of activity, because they're starting to plug these medical devices together, and they want to know up front whether they will cause damage to each other or not. So they have this project going on where they will build up a case that they can show to people, "This is how we want it delivered," and AADL plays a key role in that. Safety obviously plays a key role, and they will be folding assurance cases into it. The other interesting thing is that the Underwriters Laboratories started to get interested in getting into that business and has started to learn about that area as well.

Julien: The funny aspect is that the same technology we developed for avionics architecture, we apply that right now in the medical domain. We can think that what you want to verify and validate will be different, but in fact, it is just to make sure that the system works and to apply some certification and validation techniques.

Bill: I see. So I understand that you maintain a Wiki that's accessible to the public. What kind of information do you maintain there, and where else can people go to find more information about your work?

Julien: So we maintain a Wiki with our community that gives all the papers we have about our research on AADL. Also we have source code that is hosted on github.com. People can come here and contribute freely to the OSATE tool set that is supporting AADL. We are also open to all contributions in terms of projects that have been done with AADL, and also adding new papers about the technology. A really interesting fact is that we have new contributors in the Wiki, and they added some work we were not aware of, like work about avionics architecture in AADL that has been done in Asia with a Chinese research institute.

Peter: Some other things that you can find on the Wiki is every time we have our standard meetings, which are quarterly, we usually hold one or two days that are user days where people report out their experiences with AADL. All of those are posted on the Wiki, for example. So it's an interesting history that you can look through of people's experiences. We also have a page that gives pointers to tool sets that people have put together around the AADL that you have access for.

And finally, the Wiki is not the only source, but the main starter point source is aadl.info, our website, and from that you can get to your links to the Wikis and to all those other things. And finally, one of the things you will notice is we recently published a book on the AADL, on the use of AADL. So if you want to have that, again, go to aadl.info and you can find pointers to that as well.

Bill: Great.

Julien: And we just find that the Wiki is very useful, because we have had for many years a really active and established community. We didn't give all the tools maybe to publish necessary information. So people [have] contributed already to the Wiki and have built all the necessary documentation and all the research information that has been done during projects. So this is a really useful tool.

Bill: Peter, Julien, thanks for visiting with us today. If you'd like more information about the research in this field, you can visit aadl.info and there's a link there to the Wiki and all the other information that Peter and Julien talked about today.

You can download the SEI technical reports and notes on AADL from the SEI library website at sei.cmu.edu/library/reportspapers.cfm. This podcast is also available on the SEI website at sei.cmu.edu/podcasts and on Carnegie Mellon University's iTunesU site. As always, if you have any questions, please don't hesitate to email us at info@sei.cmu.edu. Thank you.