AdaCore

Software Safety Certification

Interview with AdaCore President, Cyrille Comar

Where did AdaCore's roots in certification stem from?

The Ada language has always had a strong presence in Aerospace and Defense, and a number of AdaCore's major customers develop software that needs to meet certification standards in this domain. In the 2003-2004 timeframe the organizations responsible for the DO-178B standard for commercial airborne software -- EUROCAE in Europe and RTCA in the US -- began an update of DO-178B that would accommodate modern technologies such as Object-Oriented Programming (OOP), Formal Methods, and Model-Based Development.

During that same period we were getting inquiries from customers who likewise were interested in adopting modern techniques such as agility and OOP for safety-critical code. In 2004 we joined the EUROCAE/RTCA working group that produced the eventual DO-178C standard and its associated supplements. On a personal note, I was on the subcommittee that produced DO-332, the DO-178C supplement on Object-Oriented Technology and Related Techniques. As a result of our participation in the DO-178C process we were able to bring our expertise in language technologies to the DO-178C working group and at the same time acquire a deep familiarity with how software is certified at the highest levels of criticality. We have subsequently leveraged this expertise in other safety-critical domains such as rail (EN 50128), space (ECSS-E-ST-40C and ECSS-Q-ST-80C) and automotive (ISO 26262 and IEC 61508), which we support through certifiable run-time libraries, qualifiable tools, and other products and services.

The Ada and SPARK programming languages are used worldwide in safety-critical domains because of the sound software engineering principles that they are based on. How does that apply to projects with certification needs?

Let me say first that safety certification applies to a complete safety-critical system such as a plane, a rocket, a train or a car, rather than to individual components. Software considerations are only a small part of the complete certification effort which also includes a global safety analysis, system architecture and design, and hardware considerations. That being said, the role of software in such systems has steadily and significantly grown, both in size and functionality, making "software aspects of certification" (the DO-178C subtitle) more challenging and important.

The major goal of such software-related standards is to enforce the application of sound engineering principles to the development of critical software; the more critical the software, the higher the degree of confidence needed that the software does what it is supposed to do, and does not do what it is not supposed to do. The major contribution that software can make to the safety of a system is to implement accurately without exceptions what is expected of it. Ada's well-defined and strong semantics, as well as its readability make it a language of choice for showing that the software implementation meets its requirements. The SPARK subset of Ada even goes one step further and makes it possible to formally guarantee the equivalence between the actual requirements formalized as mathematical properties and their implementation. Furthermore in SPARK you can prove mathematically that your software will satisfy critical security or safety properties, for example that it does not have run-time errors, thus preventing vulnerabilities like buffer overrun and integer overflow that are notorious in C and C++ code.

How are AdaCore tools designed to meet the needs of projects with the most stringent certification requirements?

Tools that are used in a certification context need to be qualifiable. This can mean different things for different certification standards but it always implies that tools must be properly designed, developed and maintained using strong and verifiable quality procedures. Among the tools that have been qualified with respect to various standards are our CodePeer vulnerability detector and code reviewer for Ada, the GNATcheck coding standard verifier for Ada, the GNATstack stack usage calculator, and the GNATcoverage source code coverage analyzer. Qualification also implies that defects in such tools should not just be fixed but should also be properly tracked and documented so that users can take appropriate mitigation measures in their own development cycle.

AdaCore's knowhow is in the combination of being able to provide the required certification standard-specific evidence of the quality of our toolset, while also issuing annual releases to incorporate improvements. Thus we can respond to customer requested enhancements while adapting our tools to a constantly changing hardware and operating system environment. We also recognize that customers with certified software may have to stay with a specific version of a tool for the duration of their project, and the Assurance edition of our GNAT Pro development environment offers a specialized service known as "Sustained Branches" to address this need, even providing maintenance updates when required to repair a critical issue.

How is AdaCore's code generation and model verification suite designed for critical systems with certification requirements?

Our QGen Model-Based Engineering toolsuite includes a code generator for a safe subset of Simulink® and Stateflow® models, producing source code in either MISRA-C or the SPARK subset of Ada. This code generator is being qualified at the highest Design Assurance Level of DO-178C (TQL1) , which means that the generated code can be trusted to preserve the semantics of the model. This code can then be compiled into an executable program that can run efficiently on embedded hardware platforms. The availability of a trusted tool that transforms models into source code is one of the key enablers for significantly shortening the development cycles of critical components. By way of background, control engineers have long been using graphical modeling languages such as Simulink to design and simulate closed-loop systems. By making their models more precise and detailed, they can now use automated code generation and avoid the expense and potential errors associated with manually coding their models. With the transformation fully trusted (as a result of the TQL1 qualification), most of the traditional certification-required verification and validation activities on the source code can be omitted or reduced, leaving more time to the development team to concentrate on the required verification activities at the model level. Not only does it significantly reduce the overall effort, but it also allows the work to be much more effective because it's done at the level of abstraction where the system has been designed. In order to accompany this shift of focus and make it even more effective, we offer a model-level verification suite including static and dynamic analysis capabilities. This toolsuite is based on technologies that have been developed and validated in the context of traditional source code verification.

What are your thoughts on the avionics industry's experience with DO-178C and on future directions for software safety certification?

DO-178C is a very mature standard that has been serving the needs of the traditional aeronautics industry; that is to say the manufacturers of large commercial aircraft and their suppliers. It has been achieving its primary goal, since very few failure incidents with commercial aircraft have been linked to software misbehaviors per se. When problems have arisen, the cause has typically been human error or a defect at the system level (for example, an incomplete safety analysis). DO-178C has been extremely effective in giving confidence that the software meets its stated requirements. Its major challenge is

that it is perceived as a fairly demanding standard and thus fairly costly to apply. It is also rooted in best software engineering practices of the end of the last century.

In actuality, DO-178C is only a fairly minor upgrade of the DO-178B standard defined in the early 1990s; it assumes a V-shaped software life cycle along with the use of 3rd generation programming languages such as C or Ada. Its main innovation is the inclusion of technology-specific supplements to deal with Object Orientation, Model-Based Development, and Formal Methods, and a domain-agnostic approach to tool qualification, but it is not designed to adapt to constantly changing programming paradigms. It would, for instance, require major updates to tackle the use of Machine Learning. It is also not sufficiently friendly or cost-effective for the new entrants in the aeronautics world, such as the future providers of autonomous flying taxis. It took a fairly long time, almost 8 years, for the industry and the authorities to agree on the fairly modest and evolutionary revisions that DO-178C brought to DO-178B.

Using a similar process to prepare the next version of the standard is thus likely to fail to address the needs of a software industry in constant and fairly rapid change. So authorities are now in the process of exploring more radical certification possibilities that would not follow the previously accepted committee-based evolution mechanism. We participate actively in the groups of experts gathered by the FAA who are working on such an initiative. In addition, and more recently, there is a strong regulatory push for software safety certification to also include cyber-security considerations. More specifically to ensure that the threat of intentional unauthorized electronic interaction to aircraft safety is correctly handled. It is recognised that this can be achieved through compliance with the jointly produced EUROCAE/RTCA specifications defining an Airworthiness Security Process, namely ED-202A, ED-203A, DO-326A and DO-356A. In this context AdaCore is part of the “High-Integrity, Complex, Large, Software and Electronic Systems'' (HICLASS) project within the UK. HICLASS is a strategic initiative to drive new, safe and cyber-secure technologies and best-practice throughout the UK aerospace supply chain. Through this working group we are able to continue contributing towards best practice, guidelines and considerations for aircraft safety certification. This is being achieved via research and development of advanced tooling to meet cyber security objectives within the security process and by showcasing how our existing technology can already satisfy cyber-security requirements.

How can AdaCore help customers meet their required certification standards?

We contribute in several ways. First, we supply specialized run-time libraries that have been used in systems certified at the highest levels of domain-specific standards in critical industries. Second, we provide development and verification tools that have been qualified for usage with respect to these standards. The certification / qualification evidence can be made available to our customers, thus simplifying their certification effort and reducing their costs. And third, services such as Sustained Branches, described above, help customers with the practicalities of software life cycle management for certified software.

Details identifying how our technology can help meet specific objectives in the various standards are provided in a series of books written jointly by AdaCore authors and external domain / certification experts; these are available for download at https://www.adacore.com/books. Current volumes in this series are AdaCore Technologies for CENELEC EN 50128:2011, AdaCore Technologies for DO-178C / ED-12C, AdaCore Technologies for Cyber Security, and, although not directly safety-related, AdaCore Technologies for FACE™ Software Developers. We are planning an analogous book for the space industry, covering standards ECSS-E-ST-40C and ECSS-Q-ST-80C.