Inside AdaCore

July–December 2017

- C Support for VxWorks and Bare Metal
- CodePeer 17 Introduces *No False Positives* Mode
- Tech Days 2017 Programs Firmed Up
- Dr. John C. Knight: An Appreciation
- Spotlighting a GAP Member: Georgetown University (Washington, DC)
- Interview with Yannick Moy
- Frama-C and SPARK Day 2017
- Check Out Recent AdaCore Blogs

C Support for VxWorks and Bare Metal

Multilanguage support has always been a major goal for the GNAT Pro technology, and several recent enhancements will be of particular benefit to C developers: both for mixed-language projects involving Ada and C, and in stand-alone C applications.

GNAT Pro now supports C development (including C11) for cross configurations, targeting VxWorks 6, VxWorks 7 and bare metal on PowerPC, ARM and x86 architectures. These toolchains provide optimal integration for mixed Ada / C applications, including compilation, debugging and automatic binding generation.

Stand-alone C development is supported in the GNAT Pro C product, currently available for VxWorks and on the roadmap for bare metal. This allows C projects to benefit from the same level of support as for Ada, and to have access to a C11 toolchain updated annually with a recent and stable version of GCC. A special service known as sustained branches is also available, allowing a C project to stabilize on a specific version of the technology while keeping the option of integrating safety-critical fixes.

As with GNAT Pro Ada, the same GNAT Pro C development environment can be used across multiple platforms. In particular, the same compiler version is provided for cross and native Windows and Linux environments, facilitating portability and lowering the risk of differences between host-based and target-based testing. Endianness portability can be handled through the GNAT "Scalar_Storage_Order" type attribute, which works the same way in C as in Ada (information about this attribute may be found in **www.adacore.com/uploads/newsletter/spring_summer_2013.pdf**).

Beyond the compiler, other elements of the GNAT Pro technology have been enhanced with C support. The GNAT Programming Studio (GPS) IDE now provides advanced C editing and navigation capabilities including auto-completion, usage-to-declaration traversal, reference searches and call graphs. The GNAT Pro C toolchain can also be driven from Eclipse, through either the Wind River Workbench environment or CDT. GNAT Project Files (.gpr) are fully C-aware and can capture both mixed Ada/C and pure C applications. For bare metal configurations, the target emulator GNATemulator is available and is supported for both Ada and C development. Depending on the target, C support is also provided in the GNAT-stack worst-case stack consumption analysis and the GNATcoverage structural code coverage analysis tools. For more information, please contact **info@adacore.com**.

CodePeer 17 Introduces *No False Positives* Mode

CodePeer 17 has introduced the switch `-messages min`, which suppresses warning messages from source program constructs where a precise analysis is not possible. This switch, enabled by default at `-level 0` and `-level 1`, lets the user control the tradeoff between detecting all potential errors and minimizing false alarms ("false positives"). The `-messages min` switch helps save development/verification effort by flagging only those constructs that are most likely to be real errors, since they are reported based on a more precise analysis of the source code. CodePeer 18, to be released during Q1 2018, will provide additional filters and heuristics to deal with the management of false alarms.

Tech Days 2017 Programs Firmed Up

Mark your calendars: AdaCore's annual customer-focused Tech Days conferences will be held this year in Paris (October 5) and Boston (November 15–16).

AdaCore technical staff will present the latest news about the company's current and planned product offerings and activities including GNAT Pro, CodePeer, SPARK Pro, and QGen. This year's events will also offer talks on code coverage approaches, dealing with platform obsolescence, GPS for bare metal development, an update on Ada 2020, and other topics. Paris attendees will hear customer presentations from David Lesens (ArianeGroup) on flight software of the Ariane 6 launcher, and Wiljan Derks (Nexperia) on Ada in a real-time Windows-based manufacturing system. Boston Tech Days will feature a keynote talk from Paul E. Black (NIST), the lead author of the NIST report *Dramatically Reducing Software Vulnerabilities*; he will summarize "lessons learned" from that report with a focus on formal methods and strong languages.

Past Tech Days conferences have been highly acclaimed by attendees; these events offer customers a unique opportunity to meet and talk with AdaCore's experts and with other users of the company's technologies. For a detailed agenda and registration information please visit **www.adacore.com/techdays/** or contact **info@adacore.com**

---

in memoriam

Dr. John C. Knight: An Appreciation

Dr. John C. Knight, Professor of Computer Science at the University of Virginia and a founder of the company Dependable Computing, passed away on February 23, 2017.

An early contributor to the Ada language design while at NASA in the late 1970s, Dr. Knight was an enthusiastic advocate of Ada and SPARK during his long and distinguished career. He was a proponent of formal methods for high-assurance software and focused on the engineering of safety-critical computer systems in domains such as aerospace, medical devices, and automotive.

Dr. Knight received the IEEE Computer Society Harlan D. Mills award (2006) and the ACM Special Interest Group on Software Engineering (SIGSOFT) Distinguished Service award (2008), and he authored the book *Fundamentals of Dependable Computing for Software Engineers* (2012).

Spotlighting a GAP Member

Georgetown University (Washington, DC)

Under the direction of Prof. J. Smart, researchers at Georgetown University used the GNAT Ada environment to develop software for analyzing highly confidential information (HIV patients' medical records) while maintaining a high level of privacy assurance (i.e., guaranteeing no possibility of exposing the data being processed). Known as ATra, the system consists of a hardware "Black Box" in which sensitive information can be input but is never allowed to be output or inspected. The device is entirely self-contained and completely automated, with no human intervention possible, and with no means to diagnose program failure or mitigate recovery. Since correctness of operation is paramount, Ada 2012 was selected as the ideal programming language for this effort, helping to ensure reliability, scalability, and portability across computer platforms that must range from an Apple macMini to Cray XC supercomputer class.

The ATra system operates by having organizations transfer encrypted data files to a set of carefully configured data folders that prevent direct access to device internals. Upon receipt, the file contents are immediately read into volatile memory, and the files are permanently and irretrievably erased. Analysis consists of detecting a specific set of patterns that all participating organizations have unanimously agreed to. These patterns are encoded in a carefully crafted algorithm within the Ada program that is securely loaded into the device. Thorough testing of the algorithm during system development uncovered an incorrect assumption about the value of one of the program variables at the beginning of a loop; this error was detected and corrected with the aid of Ada 2012 preconditions and assertions.

During the program's operation, no access to the contents of the device, or communication with the algorithm, is allowed or possible; all external interfaces as well as unnecessary operating system components and other application software are removed or permanently disabled. The only output is a set of reports that indicate the patterns that are detected.

Under a pilot program for the National Institutes of Health, the ATra techniques were successfully deployed for analysis of public health data pertaining to 168,000 individuals living with HIV in the Washington DC, Maryland, and Virginia areas. The use of ATra slashed this enormously labor intensive, multi-year process to a 24-minute computation, never exposing any private information to obtain the results. In support of the U.S. Center for Disease Control, this system is now being expanded to process data from public health organizations spanning the entire U.S. Eastern Seaboard, involving nearly 500,000 confidential patient records. The unambiguous semantics of the Ada language and its powerful programming features including robust constraint enforcement, tasking, and exception handling have enabled ATra to perform all of its necessary functions and run flawlessly. For further information please contact Prof. Smart at **smart@georgetown.edu**.

The GNAT Academic Program (GAP) is an AdaCore initiative to foster the teaching of Ada and SPARK in colleges and universities. GAP members get free access to AdaCore's GNAT technology, including support. For further information please visit **www.adacore.com/academia/projects/** *or send an email to* **gap-contact@adacore.com**.

Yannick Moy

Senior Software Engineer

Yannick, tell us about your background and how you came to be involved with Ada and AdaCore. What is your current role?

I came to software through mathematics, and my first exposure to programming was a class on algorithms and complexity using OCaml (a language we use today in the SPARK toolset) when I was 17. From the start, I loved seeing how mathematical principles related to their concrete realizations. During my studies I was lucky to have professors who bridged the gap between theory and practice, and this attracted me to the mathematical analysis of software: first at PolySpace, then Frama-C, and finally at AdaCore with my work on CodePeer and SPARK. So my job has kept improving, each time analyzing a better language, from C++ to C, then Ada, and finally SPARK! I am currently the SPARK Product Manager, and in this position I have both technical and marketing responsibilities for aligning the product with user needs. I also help select the research projects where we develop the technology that then becomes part of our tools.

The concept of formal methods in software development originated several decades ago but didn't seem to gain much traction in mainstream practice until relatively recently. What took it so long, and why the new interest?

Hardware advances have of course been a great help, and now we also have efficient tools that work on the artifact that software engineers care most about: their source code. That was not the case until recently. With formal program verification, efficiency relies primarily on the capabilities of the underlying automatic provers, and these have seen amazing gains in the last fifteen years. Couple better performance with more powerful proof engines, and the result is a progressive shift of the verification burden from programmers to tools. So much in fact that formal verification is becoming a practical adjunct to testing, even possibly replacing some of the testing in contexts such as software certification. Formal methods can provide much stronger guarantees (for example in proving the absence of run-time errors) and help reduce the cost of the overall verification effort, especially when very high quality is sought.

Another global movement that has made proof-based verification attractive in mainstream software development is the emergence of "lightweight" or "disappearing" formal methods, in which the users do not need to see the mathematical underpinnings. This is accomplished in SPARK 2014 at the language level by making contracts such as subprogram pre- and postconditions both executable and statically verifiable, and at the tool level by allowing user interactions based on the source code rather than mathematical formulas. The philosophy of lightweight formal methods is to favor gradual adoption, with a smooth learning curve and incremental benefits proportional to the effort placed in verification. This underlies our definition of a verification scale in SPARK, from Stone level to Platinum level, as explained in a booklet that we recently prepared in conjunction with our customer Thales (see **www.adacore.com/knowledge/technical-papers/implementation-guidance-spark/**).

So we now have tools like SPARK Pro that apply to source code and are usable by software engineers. This nicely coincides with an increasing interest in safety and security: more and more domains rely on complex software that is connected to the Internet, or operated and patched remotely, yet control critical aspects of our life. Why use less-than-ideal solutions based on vulnerable languages like C or C++, when languages and tools like Ada and SPARK are available and completely solve some of these problems? Based on my participation in recent conferences, I see a growing number of engineers looking for the right tradeoffs here, and formal methods are appealing to those who develop critical software.

Any hobbies or outside interests that you'd like to share?

When I want to take a break from programming languages, I turn to . . . languages! I managed to learn Italian as a hobby, and for the past few years I have been testing my slow progress in Russian with some of my AdaCore colleagues. This hobby even won me a bottle of wine once at AdaCore, as a prize for the only entry in the multi-language category of a clerihew contest! (A clerihew is a whimsical 4-line poem; see **www.verse.org.uk/what-is-a-clerihew.html** for examples.)

Frama-C & SPARK Day 2017

AdaCore was one of the organizers of *Frama-C & SPARK Day 2017: Formal Analysis and Proof for Programs in C and Ada*. This international conference, held in Paris on May 30, brought together Frama-C and SPARK/Ada technology developers and users to compare their formal program verification approaches and to highlight recent progress and experience. Talks on SPARK usage included a verified CubeSat operating system (Vermont Technical College, US), a flight stack for a high-altitude glider (Technical University München, Germany), and high security components running on top of a separation kernel (secunet, Germany).

A presentation from Thales, France, described that company's process of adopting the SPARK technology; this is further detailed in a booklet *Implementation Guidance for the Adoption of SPARK* that was co-authored by AdaCore and Thales and distributed to conference attendees.

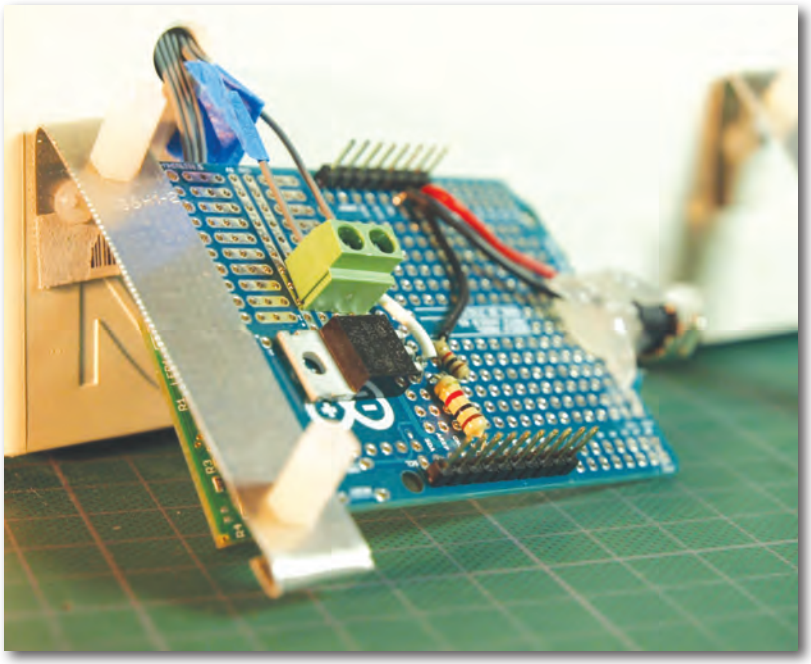
The conference attracted around 120 participants, evenly split between academia and industry, and plans are in place for making it an annual event. For further details about this year's presentations please visit **www.open-source-innovation-spring.org/frama-c-spark-day/**.

Yannick Moy photograph by Gary Matoso

Check Out Recent AdaCore Blogs

Here are highlights from some of AdaCore's blog entries that have appeared during the first few months of 2017; for the full articles please visit **blog.adacore.com**.

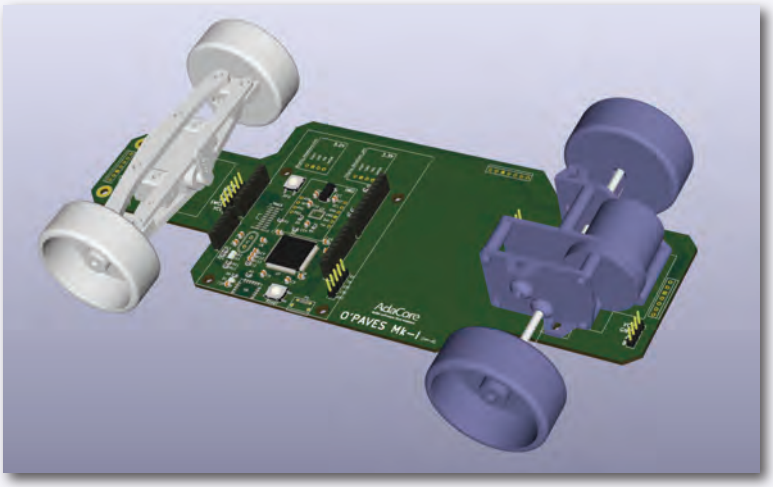
- ***DIY Coffee Alarm Clock***, by Fabien Chouteau, shows how Ada software for an STM32F469 board can add an alarm clock interface to a commercial espresso machine (see photo, right). The software components include a user interface through the screen on the STM board (to display the time, and to allow setting the alarm) and a control function to signal the espresso machine when to turn on. This "Do It Yourself" project is intended as a proof-of-concept, illustrating how to use Ada in an embedded system.
- ***Introducing Libadalang***, by Raphaël Amiard and Pierre-Marie de Rodat; ***Going After the Low Hanging Bug***, by Raphaël Amiard, Yannick Moy and Pierre-Marie de Rodat; and ***(Many) More Low Hanging Bugs***, by Yannick Moy. These articles describe the new Libadalang technology for program analysis and its application in multiple lightweight checkers for possible quality issues in Ada source code. This type of analysis can be a useful part of code review and is planned for inclusion in a future release of CodePeer.
- ***GPS for bare-metal developers***, by Anthony Leonardo Gracio, explains how to use GPS to achieve the bare-metal "holy grail": Build, Flash, Debug. GPS now supports two different approaches for connecting to a remote board: the ST-Link utility tools for STM32-based ARM boards, and OpenOCD which supports ARM boards and probes using a JTAG interface.



Arduino proto shield for Do-It-Yourself Coffee Alarm Clock. Photo: Fabien Chouteau

- ***Simics helps run 60 000 GNAT Pro tests in 24 hours***, by Jérôme Guitton, Jérôme Lambourg and Joel Brobecker, summarizes how AdaCore uses Wind River's Simics simulator to handle the daily GNAT Pro test runs on the large number of supported VxWorks configurations. Simics provides a stable framework with efficient test execution and expedites the QA process.
- ***SPARK Tetris on the Arduboy***, by Fabien Chouteau, Arnaud Charlet and Yannick Moy, describes the port of the Tetris videogame, written in SPARK, to the Arduboy platform. The Arduboy uses an 8-bit AVR microcontroller, which is not one of the GNAT targets, so building an executable was a two-step process: compiling the SPARK code into C, and then using the C compiler from the Arduino toolchain to generate code for the Arduboy board.
- ***Getting started with the Ada Drivers Library device drivers***, by Pat Rogers, introduces the drivers and demonstrations that are included in the Ada Drivers Library; this library comprises Ada device drivers and examples for ARM-based embedded targets and is available on GitHub. The demonstrations can serve as starting points and as API references when incorporating devices into client applications.

Topics covered by other blogs include a student project in a real-time systems course at the Polytechnic University of Valencia, Spain; a Libadalang-based tool for detecting copy/paste instances; and an efficient and expressive strings package in GNATCOLL.



O'PAVES vehicle frame. Photo: Fabien Chouteau

Autonomous-Vehicle Prototyping Platform Nearing Delivery

Project O'PAVES (Open Platform for Autonomous VEhicle Systems), an effort funded by the European Union through the CPSE Labs consortium, is on track to deliver a complete open-source platform for prototyping autonomous vehicle systems. Scheduled for release on GitHub in October 2017, the platform will consist of hardware, software and development tools. The hardware is designed to be affordable and easy to build: the main element is a printed circuit board that serves as both the electronics and the vehicle frame (see photo, left), and the other parts are either off-the-shelf or 3D printed. All software components and tools will be provided by AdaCore as Freely Licensed Open Source Software (FLOSS). For further information please visit **hackaday.io/project/17555-opaves**.

Continuing GNAT Pro Support for FACE

The Future Airborne Capability Environment (FACE) is a government/industry open-standard initiative to promote robustness, interoperability and portability for military avionics applications. AdaCore has been actively participating in the FACE Consortium since 2012, to help ensure that Ada's software engineering benefits are appropriately reflected in the Ada language profiles. Most recently, AdaCore has started working with an approved FACE Verification Authority to verify that GNAT Pro Ada meets the applicable FACE Technical Standard requirements for the Ada Language Runtime Safety and Security Profiles. Other FACE-related activities include the Q1 2017 release of the GNAT Pro 17.1 Ada Development Environment for Wind River's FACE Certified VxWorks 653 Platform 2.5; VxWorks 653 is the first Commercial-Off-the-Shelf product to be certified as conformant to the FACE Technical Standard's Operating System Segment (OSS) Safety Base Profile.

QGen Selected by MASC for Avionics System

MHI Aerospace Systems Corporation (MASC), a member of the Mitsubishi Heavy Industries Group, has selected the QGen toolset to develop the software for the Throttle Quadrant Assembly (TQA) system. This avionics research project is being conducted to meet the Level C objectives in the DO-178C safety standard for airborne software and its DO-331 supplement on Model-Based Development and Verification. A qualified tool can help save significant effort in developing and verifying the software, and AdaCore's planned qualification for the QGen code generator at Tool Qualification Level TQL-1 (equivalent to a development tool in DO-178B) factored strongly in MASC's decision to choose QGen.

SPARK Adoption Guidelines Published

A free booklet co-authored by AdaCore and Thales shows how best to introduce and make use of the SPARK/Ada formal verification technology based on a project's assurance goals. *Implementation Guidance for the Adoption of SPARK* describes the various levels of software assurance where the SPARK language and toolset can be used, including Bronze (proving proper initialization and data flow), Silver (proving absence of run-time errors), and Gold (proving key integrity properties). The booklet explains the benefits and costs at each level and details the processes that Thales is using to introduce formal verification in operational projects. It is available for free as a download from **www.adacore.com/impl-guidance-spark/** or as a printed copy by request to **info@adacore.com**.

AdaCore / DO-178C Booklet Published

A new booklet, *AdaCore Technologies for DO-178C / ED-12C*, is now available from AdaCore. Written by certification expert Frédéric Pothon and AdaCore's Business Development Lead Quentin Ochem, the booklet explains how to use the Ada language and AdaCore's tools and run-time libraries to help meet the objectives in the suite of software certification standards for commercial airborne software. The booklet covers DO-178C and its associated Tool Qualification Considerations (DO-330) and technology supplements: Model-Based Development and Verification (DO-331), Object-Oriented Technology and Related Techniques (DO-332), and Formal Methods (DO-333). The booklet is available for free as a download from **www.adacore.com/tech-do-178c/** or as a printed copy by request to **info@adacore.com**.

calendar highlights / July–December 2017

For up-to-date information on conferences where AdaCore is participating, please visit **www.adacore.com/events/**.

GNU Cauldron 2017
September 8–10, 2017 / Prague, Czech Republic

AdaCore is a major sponsor of this event.
gcc.gnu.org/wiki/cauldron2017

AdaCore Tech Day EU
October 5, 2017 / Paris, France

For details on this annual customer-focused event please see the companion article in this newsletter.
www.adacore.com/techdays/eu

ARM TechCon
October 24–27, 2017 / Santa Clara CA, US

AdaCore is a sponsor and will be exhibiting at this event.
www.armtechcon.com/conference/

HIS 2017 (High Integrity Systems Conference)
October 17, 2017 / Bristol, UK

Ada Core is a major sponsor and will be exhibiting at this event.
www.his-2017.co.uk

RSSRail 2017 (Reliability, Safety and Security of Railway Systems Conference)
November 14–16, 2017 / Pistoia, Italy

Eric Perlade will present a tutorial, "AdaCore Technologies for EN 50128:2011".
conferences.ncl.ac.uk/rssrail/

AdaCore Tech Days US
November 15–16, 2017 / Boston (Burlington) MA, US

For details on this annual customer-focused event please see the companion article in this newsletter.
www.adacore.com/techdays/us

Public Ada Training
November 27–December 1, 2017 / Paris, France

AdaCore is conducting an Ada Fundamentals course, which combines live lectures with hands-on workshop exercises.
www.adacore.com/public-ada-training

ESE Kongress 2017 (Embedded Software Engineering)
December 4–8, 2017 / Sindelfingen, Germany

AdaCore will be exhibiting at this event.
www.ese-kongress.de/english/

ESC Silicon Valley (Embedded Systems Conference)
December 5–7, 2017 / San Jose CA, US

AdaCore will be exhibiting at this event.
escsiliconvalley.com/

Inside AdaCore is published twice a year simultaneously in New York and Paris by AdaCore.

150 W. 30th Street, 16th floor
New York, NY 10001, USA
tel +1 212 620 7300
fax +1 212 807 0162

46 rue d'Amsterdam
75009 Paris, France
tel +33 1 49 70 67 16
fax +33 1 49 70 05 52

info@adacore.com
www.adacore.com

AdaCore