AdaCore WHITE PAPER

Why use a commercially supported C/C++ toolchain?

AdaCore recently extended its support of C and C++ toolchains and now supports not only native environments (such as Linux or Windows) but also cross ones such as VxWorks, embedded Linux, and even bare metal (C only at the time of this writing). In a world where most developers have access to a C/C++ toolchain that's included with hardware or OS environments, this produces a question - what's the added value of a commercially supported compiler for C and C++?

A qualified toolchain for industrial usage

Building and qualifying a toolchain for industrial usage involves more work than it may appear at first. At AdaCore, our compilers are GCC-based. You might think that all we need to do to provide a GNAT Pro compiler is fetch GCC sources and compile and package it. However, it actually takes us about a year between the day we obtain a build from the open-source repository and the day when we stabilize it and manage to make it pass our quality assurance suite, which includes 25 years of accumulated test cases from the industry.

We aren't saying the work done by the community isn't of high quality - it most definitely is a fantastic achievement. However, qualifying the toolchain we provide to customers requires making it pass tests cases that couldn't be shared on an open-source platform because they include confidential code, sometimes entire customer applications of over beyond a million lines of code, stressing the toolchain in unique ways that come directly from the field, which are sometimes specific to environments not commonly available.

Using an industrial toolchain means that you're using a toolchain that has been built and verified with stringent industrial requirements in mind, and is used and tested by other industrial users with similar needs, all the more likely to confirm its appropriateness. But there's more.

A toolchain with support provided from the experts

Compilers are extremely complex programs that constantly evolve to support new language features, new optimizations, and new instruction sets. In this context, it's not uncommon to hit roadblocks when using them. This may be due to toolchains that are difficult to configure (think for example of the hundred of switches typically available for the compiler and linker), features that need tweaks to be usable, or even on rare occasions plain old bugs.

A good order of magnitude estimate cost of an engineer in an industrial context is about $1,000 / day. Having one or more l engineers blocked on compiler problems can quickly translate into 5 or 6 digits of costs while forcing them to investigate problems in areas in which they don't have the expertise (e.g., optimization technology). Having a professional support contract is an assurance that should any problems occur, they'll be quickly resolved . In particular, in the case of AdaCore, support is provided directly by toolchain developers. If it's a configuration issue, they'll be able to explain the nuances of the tool and offer alternatives. If it's a feature missing or that needs tweaking, they'll be able to modify the tool to make it meet the need. If it's a plain bug, they'll be able to fix it and provide a working toolchain in a matter of weeks, sometimes days, all the while providing fast workarounds so that the team can continue working while waiting for resolution.

Stability over decades

Industrial systems often require to set a toolchain for years - sometimes decades - all the while continually monitoring for potential issues and maintaining the possibility of getting minimal fixes for critical problems. Products such as GNAT Pro Assurance provide exactly that. Projects can select a specific version of the technology and receive support services on that version for many years. While keeping the underlying tools extremely stable, maintenance includes potential fixes for critical problems by allowing minimal tool upgrade in case of a serious issue.The project can then decide to adopt the minimally fixed toolchain to deploy a new version of the application or

just to compare newly compiled binaries with the original ones to identify whether these fixes (and therefore the initial problem) have an impact, and if yes, where.

This provides strong assurance of stability over the lifetime of the application, including after its deployment and again ensures that whatever issue could arise, there will be a set of experts available to provide insights and options.

Certification artifacts

In the most stringent safety-critical context, having a supported toolchain isn't sufficient. Certification standards require demonstrating that the toolchain is usable in a certified context. What this demonstration entails varies depending on the industry. All require the kind of long-term maintenance described in the previous section, all require that the, the language library (also known as standard library or run-time) to be certified. Each standard has specific additional requirements: Avionics (DO-178) requires source to object traceability analysis at the highest level, Rail (EN-50128) and Automotive (ISO-26262) require compiler qualification, etc.

Certification artifacts typically include development processes evidence, quality assurance, and verification documents, which can be extremely costly to develop independently for a specific project. It may indeed not be possible without deep insights into the technology. Using a commercially supported toolchain is also strong assurance that, should such a need arise, projects are already using a technology for which AdaCore can provide these artifacts .

Cross-platform portability

Another key element in selecting an independent vendor toolchain is ensuring portability as things evolve over time. Indeed, a toolchain coming with a specific environment may have specificity to this environment. This can cause difficulties when it comes time to migrate to another hardware or OS vendor. For example, one vendor might provide an old custom GCC build, another an LLVM build, and a third one enable specific language extensions or custom run-times,. This can become a serious problem in situations where the same code needs to run on several platforms at the same time, for example both on a Windows or Linux test-bed together along with the target cross environment, or, as is often the case with product-line driven strategies, on different targets at the same time.

A vendor such as AdaCore literally supports hundreds of different environments, each of which is a combination of different hardware and operating systems. There may also be multiple variants within a specific hardware/software system. All of these are all targeted by the same toolchain. This provides a very strong assurance of the ability to preserve a software investment over time, ensuring that adopting new environments is as easy as possible, and increasing the number of ways to optimize market positioning of industrial applications.

Beyond the toolchain

Last, but not least, toolchains are more than a compiler / linker / assembler / debugger suite. Additional tools are often provided to complete the development package. In the case of GNAT Pro, these include an IDE, a bare-metal emulator, an advanced multi-language builder, and a static stack usage checker. All these tools are provided with the same level of quality and support as the compiler itself.

Conclusion

Today's world of toolchains is full of options and it's been decades since it was last necessary to spend thousands of dollars to be able to compile a first "hello world". This is a good thing, which has enabled generations of developers to learn valuable techniques and start projects at minimal costs. The good news is that while these commodity compilers are widely available, some companies such as AdaCore are positioned for the next step, when applications require industrial-grade guarantees and services.

AdaCore