Heziode

AdaCore Tech Day 2019

26 novembre 2019

Table des matières

Le 3 octobre 2019 s'est tenu l'AdaCore Tech Day à Paris. Durant cette journée, AdaCore a présenté ses nouveautés, ce qui était en cours de développement et leur feuille de route pour 2020. AdaCore est une entreprise leader dans le support du langage de programmation Ada, qui est très utilisé dans le développement de logiciels critiques.

Cet article a pour but de présenter les principales nouveautés qui ont été exposées durant l'AdaCore Tech Day ; il ne se veut ni exhaustif, ni détaillé.

i

Je tiens à préciser qu'au moment où j'écris ce billet, je ne travaille pas pour/chez AdaCore. Je n'ai aucun lien directe avec eux, je ne possède pas d'action chez eux, et je ne suis pas sponsorisé/affilié/`^(?:\w+é)$`.
En revanche, je suis un utilisateur de leur compilateur.

1. Rappel sur Ada et Adacore

1.1. Ada

Ada est un langage de programmation qui représente l'aboutissement de la lignée des langages «classiques», impératifs et procéduraux. Il constitue essentiellement un effort de synthèse des meilleurs éléments figurant dans les langages qui l'ont précédé, intégrés dans un ensemble cohérent. Il est utilisé avec succès dans des domaines aussi variés que le temps-réel, la gestion, la CAO, le médical, le traitement linguistique…

Ada a été conçu d'après un cahier des charges dont l'idée directrice est de diminuer le coût des logiciels en tenant compte de tous les aspects du cycle de vie. Le langage est donc bâti autour de quelques idées-force :

— privilégier la facilité de maintenance sur la facilité d'écriture, car la maintenance représente près des $\frac{2}{3}$ des coûts d'un logiciel ;

— fournir un contrôle de type extrêmement rigoureux, permettant de diagnostiquer les erreurs le plus tôt possible ;
— permettre une programmation intrinsèquement sûre, en permettant au logiciel de traiter toutes les situations anormales ;
— être portable entre machines d'architecture différentes, afin de ne plus lier les logiciels à un constructeur ;
— permettre des implémentations efficaces et donner accès à des interfaces de bas niveau, indispensables à la réalisation de systèmes «temps réel».

Ada est une norme internationale (ISO/IEC 8652). Tous les compilateurs actuellement sur le marché ont été validés selon une procédure extrêmement rigoureuse qui assure leur conformité à la norme. Aucun sur-ensemble ni sous-ensemble n'est admis, afin de garantir la portabilité des applications.

Ada a été le second (précédé de quelques mois par Common Lisp) langage orienté objet normalisé au niveau international. Ils sont restés les seuls jusqu'à la normalisation de C++ trois ans et demi plus tard (ISO/CEI 14882 :1998)

1.2. AdaCore

Depuis 1994, AdaCore développe GNAT, l'implémentation d'Ada pour GCC, à la base initiée par la NYU (New York University). En 25 ans, AdaCore a réussi à se hisser au sommet du marché en demeurant aujourd'hui le seul fabricant de compilateur Ada à fournir une implémentation complète du standard Ada dans sa dernière version, ainsi que toutes les annexes.

En plus de rajouter des supports d'architectures, AdaCore a également développé tout un ensemble d'outils gravitant autour du compilateur, servant à améliorer la qualité de code et à aider à la certification ferroviaire, aéronautique et spatiale pour laquelle il dispose d'un département dédié.

La vision d'AdaCore est d'aider les gens à concevoir des logiciels de très grande qualité.

Malheureusement, comme Ada est un langage utilisé dans un secteur de niche (ferroviaire, aéronautique, spatial, etc.) et qu'il a du mal à se faire connaître du grand public, il était indispensable de rendre accessibles des bibliothèques écrites dans d'autres langages. Ainsi, AdaCore met un point d'honneur à fournir un interfaçage avec C et C++ le plus complet possible.

2. Amélioration de l'écosystème Ada

2.1. GNAT LLVM

Lors de la conférence, AdaCore a annoncé un support de LLVM pour le compilateur GNAT! De nombreux projets de compilateurs s'intéressent à LLVM afin de profiter de l'écosystème offert par cette infrastructure de compilation (optimiseur, machine virtuelle, etc).

Le port n'en est qu'à ses débuts, mais il est déjà fonctionnel et dispose d'informations de débogage minimales. Le projet n'est cependant pas entièrement mature et n'est conseillé que dans le cadre de projets de recherche.

Des investigations sont également prévues pour intégrer le support de KLEE (moteur d'exécution de l'écosystème LLVM) et le générateur de code GNAT CCG. Vous pouvez retrouver GNAT LLVM sur leur GitHub . AdaCore a tout de même souligné que cela ne changerait strictement rien à leurs offres GNAT Pro qui resteront sur GCC.

2.2. libadalang, le remplaçant d'ASIS

AdaCore a également annoncé qu'après GNAT Pro 20, ASIS (Ada Semantic Interface Specification) sera mis en version *standalone* et AdaCore n'envisage pas d'y rajouter Ada 202X. libadalang est une librairie qui permet de manipuler un code source Ada. Elle fournit une API de haut et de bas niveau permettant d'interroger et de modifier le code source.

Voici des exemples de requêtes auquel peut répondre libadalang:

— Quel est le type de cette expression?
— Combien y a-t-il de référence à ce type?
— Donne-moi l'emplacement dans le code source de ce jeton.
— Renomme-moi cette entité.

Elle fournit un support multilangages et permet ainsi de générer des *bindings* pour Ada, C et Python. En plus de cela, elle est facilement scriptable, en permettant de créer rapidement et de façon interactive des prototypes.

Au niveau des avantages par rapport à ASIS, AdaCore distingue deux aspects.

D'un point de vue implémentation, elle fournit un meilleur support pour les erreurs, des versions incrémentales (pas besoin de tout recompiler), une tolérance aux fautes et au code incomplet.

Du côté de l'API, elle fournit une API de haut niveau, permet un *binding* vers d'autres langages et permet également la réécriture de l'arbre.

libadalang, qui se veut donc être le remplaçant d'ASIS, continuera son expansion avec l'intégration dans plus d'outils de l'entreprise (GNATtest, Ada Web Server, GNATcheck, etc.). Leur feuille de route est axée sur l'intégration aux IDE, notamment sur la complétion, la recherche des références et le *refactoring*. Ils souhaitent également fournir plus de requêtes d'analyse, le support de Ada 202X et améliorer la réécriture de l'arbre.

2.3. Refonte de l'IDE

AdaCore a annoncé que GPS (GNAT Programming Studio), devient GNAT Studio. Outre le changement de nom, un important changement a été fait au niveau de son fonctionnement interne.

2.3.1. Passage à LSP

En effet, il migre petit à petit sur LSP (Language Server Protocol), un protocole créé par Microsoft pour faciliter le support des langages dans les éditeurs de code. LSP est une technologie multi-plateformes qui se répand de plus.

LSP permet de fournir le support d'un langage sous forme d'un « serveur » qui peut être utilisé par un IDE (ou simple éditeur de code) qui le supporte. Officiellement, AdaCore développe et supporte le LSP Ada (et SPARK) pour GNAT Studio, mais un plugin VSCode est dors et déjà disponible et fourni par eux-mêmes. De plus, le LSP Ada (comme tout autre LSP) peut être supporté par n'importe quel IDE, du simple éditeur de code en ligne de commande tel que, vim, emacs, ou encore en éditeur graphique avec Sublime Text ou Atom ou directement avec les IDE de Jetbrains (IntelliJ entre autres).

Le LSP Ada (et SPARK) , qui utilise libadalang, implémente pour l'instant toutes les requêtes liées à la navigation et aux boîtes à outils. Pour l'année prochaine, ils ont prévu d'ajouter le support de l'auto-complétion, le formatage et les diagnostics.

2.3.2. Quoi de neuf pour les utilisateurs avec ce nouvel IDE ?

Ce « nouvel » IDE, qui sera disponible dès la version 20, apporte des info-bulles et menus contextuels revisités pour fournir une meilleure expérience utilisateur.

On notera de nombreux changements :

— plus besoin de recompiler pour avoir les cross-références
— plus rapide à démarrer
— meilleure gestion des *separate*
— LSP donne la possibilité d'utiliser d'autres éditeurs de codes pour développer en Ada
— bien sûr, il est open source et est disponible sur GitHub.

Pour le futur, il est bien évidemment prévu d'implémenter toutes les requêtes du LSP, d'utiliser le LSP avec GNATbench, de migrer les plugins vers Python 3, de fournir le support d'autres LSP dans l'IDE (C/C++ et Python notamment). Enfin, ils envisagent également de jeter un coup d'œil au Microsoft Debug Adapter Protocol.

3. L'apprentissage d'Ada et SPARK

Comme dit en introduction, Ada (et SPARK) sont dans un secteur de niche, et ont beaucoup de mal à s'en extirper pour toucher d'avantage de monde.

AdaCore a mis en place un site permettant d'apprendre Ada et SPARK, sans rien avoir à installer sur sa machine. Ce site se nomme learn.adacore.com . Il fournit tout un ensemble de ressources pour apprendre Ada et SPARK et dispose également de chapitres pour les développeurs qui viennent de C++ et Java.

AdaCore fournit également un ensemble de livres destinés à l'apprentissage d'Ada et SPARK, ainsi que des livres axés sur la certification et la cybersécurité.

AdaCore dispose d'un programme académique pour aider les professeurs enseignant Ada. Le programme académique offre un accès à GNAT Community, des cours exclusifs et un support professionnel. Il est aujourd'hui utilisé dans 31 pays, 136 universités et compte 507 membres.

Enfin, une nouvelle édition de Make With Ada a démarré ; il s'agit d'un concours de programmation embarquée en Ada ou SPARK visant à promouvoir le langage. Les 10 finalistes se verront offrir une récompense.

De nombreuses nouveautés ont été présentées cette année, notamment avec une libadalang en pleine expansion offrant de nombreuses possibilités pour les outils manipulant du code éventuellement non compilable (IDE, etc).

Cet article n'a cité qu'un petit ensemble de ce qui a été annoncé. Pour plus d'informations, vous pouvez retrouver tous les supports des présentations sur le site d'AdaCore .

N'hésitez pas non plus à jeter un œil du côté de leur GitHub , où ils ont pas mal de projets open-source.

La miniature du tuto vient de GetAdaNow avec l'autorisation de l'utiliser « pour tout ce qui concerne Ada ».

Liste des abréviations

ASIS Ada Semantic Interface Specification. 1, 3

CCG Common Code Generator. 3

GCC GNU Compiler Collection. 2, 3

GNAT Gnu Ada Translator. 1–5

GPS GNAT Programming Studio. 3

IDE Integrated development environment. 1, 3–5

LLVM Low Level Virtual Machine. 1–3

LSP Language Server Protocol. 4