The Ada Computer Language

Its History and Evolution

by Ann Gordon

Table of Contents

The Ada Computer Language

Its History and Evolution

I. Introduction

Ada is a programming language originally developed for the military. A high-level language known for its reliability, Ada is of special value in the development of large programs, like systems for government agencies. Considered a rich and powerful language, "Ada is the best language to use for systems requiring real-time processing and high reliability," said Barry Boehm, director of the Center for Software Engineering at UCLA [Anthes].

Ada is not a superset or extension of any other language. It does not allow the dangerous practices or effects of old languages; however, it does provide standardized mechanisms to interface with other languages such as FORTRAN, Cobol, and C [Kempe].

II. Ada Language Origin

In the 1960s and 1970s, the US Department of Defense (DoD) used more than 1,500 languages for its programming. Most of these languages had been developed for one specific job and were not used for other programming tasks.

Development: HOLWG

In 1975, the DoD formed the High-Order Language Working Group (HOLWG) in search of a language that could be used for just about anything, including:

- systems programming
- artificial intelligence
- real-time programming
- embedded systems

This group called upon international development teams to help design a language that would meet the requirements issued by the US Department of Defense. HOLWG leaders even made a contest out of the endeavor [DeArment]. All "teams" created Pascal-based languages. In 1979, the language ultimately chosen was created by a Honeywell-Bull team in France.

Name: Lady Lovelace

In 1979, the DoD completed documentation on the language that had been dubbed "Ada" and the language was standardized in 1983.

This versatile language had been named Ada to honor the 1800s mathematician and scientist, Lady Ada Lovelace (1815-1852), daughter of Lord Byron. Lady Ada envisioned that mathematics would eventually develop into a system of symbols that could be used to represent anything in the universe. "Lady Ada is often considered to be the world's first programmer" [McCormick].

III. Ada Language Intent

Ada proved to be a good software engineering language that was easy to maintain. The original language, Ada 83, was controlled by the DoD; anyone outside the DoD had to secure authorization from the Defense Department in order to create an Ada compiler. The US Ada mandate (Public Law 101-511 Sec 8092) reads:

> Notwithstanding any other provisions of law, after June 1 1991, where cost effective, all Department of Defense software shall be written in the programming language Ada, in the absence of special exemption by an official designated by the Secretary of Defense.

According to a Secretary of Defense memo dated August 1984, "Ada is not only a facilitator of software engineering best practice, but also has inherent features which uniquely support both real-time systems and safety-critical systems" [Kempe 3.13.2]. This letter, addressed to all Secretaries of Military Departments, also emphasized the importance of software reuse, stressing the point that using Ada would facilitate that cost-reducing functionality.

Despite the mandate, however, relatively few programmers used Ada, and the mandate was eventually dropped in 1997. Developers and engineers claimed and Ada was difficult to use.

IV. Evolution of Ada

Standardization

In 1983, Ada was standardized by the American National Standards Institute (ANSI) and in 1987 it was standardized by the International Standards Organization (ISO). "Although Ada was originally designed to provide a single flexible yet portable language for real-time embedded systems to meet the needs of the US DoD," its domain expanded to include areas such as distributed systems and scientific computation [Guerby p.1]. Its user base expanded to include all major defense agencies in the West, the aerospace community and other sectors such as telecommunications and monitoring systems.

Revision Committee

In 1988, the Ada Board started a revision process for Ada 9X. Close consultation with the committee at ISO was important for ensuring that the needs of the whole Ada community were taken into account and not just the needs of the defense community [Guerby p.3]. The community of users submitted more than 750 revision requests, which then became a detailed set of requirements. This requirements document outlined 41 Requirements and 21 Study Topics related to Ada's revision, stressing that "support for reliability, safety and long-term maintainability should take precedence over short-term coding convenience" [Guerby p.4].

The Ada 9X Design Team, composed of consultants from all over the world, was based at Intermetrics, Inc [Kempe]. The Ada 9X process was very open, and many documents can still be downloaded from ftp://sw-eng.falls-church.va.us/public/AdaIC/pol-hist/history/9x-history. Although the Ada 9X Project Office has been closed, it is assumed by many that the next Ada revision will occur around the year 2005.

Revision Areas

Several of the main areas needing upgrade and revision were [Anderson]:

- Interfacing - including interfacing into a GUI system
- Programming by Extension - related to reusability
- Program Libraries - involved changing Ada's flat library format to one more flexible
- Tasking – adding a more static approach for common shared-data access applications
- Support – support for international character sets

Ada 95

Ada 95 is noted for having many strengths, one of them being its efficient and strong support for software fault tolerance. Those familiar with Ada 95 often tout its worthiness as a programming language. Peter Coffee wrote, "Any shop looking for a nonproprietary object-oriented language will find Ada 95 an attractive alternative to Smalltalk, C++ or Java" [Coffee 1995]. Furthermore, if users choose to work with ObjectAda 7.0 for Windows, they get the option of Java applet deployment.

V. Expanded Use of Ada

Release to the Public

In 1987, the DoD released Ada to the public and by 1990, "over 200 validated Ada compilers had been produced" [DeArment]. Ada has been used in a wide range of applications: banking, medical devices, telecommunications, air traffic control, airplanes, railroad signalling, satellites, rockets, etc. At the time of its release, Ada embodied "features to encourage modern design and modularization" [Kemerer p. 6].

Compilers

A complete GNU compilation system (GNAT) became available in 1993. The GNAT accepted Ada 95 source code and generated executable (machine) code. GNAT generated code and supported tasking for many computer platforms, including SUN Spark, UNIX and Next, although it did NOT support tasking on MS-DOS. For tasking with GNAT and a PC platform, users were advised to consider using other operating systems such as Linux or OS/2 [Kempe 4.2.1].

Hundreds of Ada compilers have been on the market, such as ActivAda and ObjectAda from Thomson Software Products and OpenAda from Rational Software Corporation. You can read about Ada compilers on this Web page: http://www.adahome.com/FAQ/comp-lang-ada.html#compilers.

For a comprehensive list of free Ada compilers, consult the following Web site: http://www.adahome.com/Resources/Compilers/Free.html. Most newer Ada compilers are backward compatible.

Education

Ada is also recognized as an excellent vehicle for education in programming and software engineering, including for a first programming course.

Since 1995, education versions of Ada (called Ada/Ed) have been freely distributed for use by educational institutions. These versions are compatible with Macintosh, DOS and Windows machines. This project was originally sponsored by The George Washington University and the United States Advanced Research Projects Agency [Kempe 4.2.3].

You can download Ada reference manuals from Simtel at http://www.simtel.net/. Just type Ada in their Search field to display a list of their Ada resources.

VI. Ada Today

Ada 95 was listed among the languages that put "the greatest emphasis on getting the same behavior from the same source code, regardless of platform" [Coffee 1996]. Other languages in this category were FORTRAN 90, Java, ANSI C and ANSI Common Lisp.

Although some people think Ada is antiquated and has disappeared by now, that is not the case. In certain communities, particularly aviation, Ada has remained the programming language of choice. Ada advocates claim that the decades-old language can solve a few of today's most pressing problems – those of security and reliability.

“We’re seeing a resurgence of interest,” said Robert Dewar president of AdaCore. “I think people are beginning to realize that C++ is not the world’s best choice for critical code” [Jackson].

VII. Conclusion

Perhaps the largest complaint programmers have against Ada involves the difficulty they face getting a program through the compiler. Ada compilers check a lot of stuff, and they will catch just about anything that could cause the slightest problem. While C++ compilers will accept pretty much anything, this means programmers and testers will spend more time debugging later. Since most Ada problems are caught right up front by the compiler, once the program passes this test, it works. This reliability assures Ada’s continued use.

And who still uses Ada? Not surprisingly, the DoD still uses Ada along with NASA. Ada supporters believe that the security and dependability offered by this program make it perfect for code that absolutely must be correct.

> Programs written in an extension of Ada, called Spark, will be used to run the next generation U.K. ground station air traffic control system, called Interim Future Area Control Tools Support (IFacts) [Jackson].

It appears that programmers can expect to see more of Ada in the future.

References

Anderson, Christine M. "Foreword: Ada 95 Rationale," January 1995. lglwww.epfl.ch/Ada/rat95

Anthes, Gary H. "Defense Department backs off Ada Mandate," Computerworld, November 11, 1996, 60.

Coffee, Peter. "An Armistice in Language Wars," PC Week, December 23, 1996, 68.

Coffee, Peter. "Java just one of many Portable App Languages," PC Week, January 13, 1997, 48-49.

Coffee, Peter. "*Unix Haters* sheds light on System Flaws," PC Week, January 23, 1995, 24.

DeArment, Eric. "Introducing Ada," January 19, 1998. www.adahome.com

Deer, Louise. "Cmgt576 Lecture," University of Phoenix Online program, June 13, 1998.

Jackson, Joab. "The Return of Ada," Government Computer News, April 14, 2008. http://www.gcn.com

Kemerer, Chris F. Software Project Management: Readings and Cases. IRWIN/McGraw-Hill, 1997.

Kempe, Magnus. "Frequently Asked Questions," Ada 95 Hypertext Reference, DocAda™. http://www.adahome.com/FAQ

Guerby, Laurent. "Ada 95 Rationale," January 1995. lglwww.epfl.ch/Ada/rat95

McCormick, John. "Happy Birthday Ada," December 10, 1997. http://www.adahome.com

Wong, Sy. "Long Overdue Unified Hardware/Software CDL Comes to Light," Electronic Design, May 13, 1998, 60.