Promoting Ada at the National Security Agency

Pamela S. Kimminau and Russell J. Graves

Department of Defense
National Security Agency
Fort George G. Meade, MD 20755-6000

Abstract

Similar to the barriers faced by Ada proponents in commercial industry and other branches of the Department of Defense, we have faced many difficulties in adopting Ada at the National Security Agency (NSA). The difficulties encountered when introducing Ada, though often proclaimed to be technical problems, actually stem more from people's unfavorable perceptions of the language. Hidden behind technical excuses, the negative attitudes towards Ada are a resistive barrier to the spread of Ada's use. This paper identifies some of the technical and attitudinal barriers that exist at NSA, attempts to explain why they exist, introduces some methods being used to diminish or eliminate the barriers, and discusses previous and current Ada development projects.

Introduction

The introduction of Ada at the National Security Agency (NSA), just as within most of the software development world, has been slow and faced its share of obstacles. From the mountains to the molehills, we have had to tackle the common excuses and genuine problems posed by developers and managers alike. From the C fanatics to the "old dogs", Ada has faced opposition. To overcome some of the obstacles and opposition, we formulated a two-part strategy -- identify the barriers and deal with them the best way possible. After asking numerous questions (regarding the lack of Ada development efforts at NSA) and receiving many unjustifiable answers, we determined that the barriers stem more from the attitudes people have formed towards Ada than from actual technical problems they encountered using Ada. The technical barriers were already being addressed, the impact of adverse attitudes or how to handle them had not been adequately considered. Thus, we were back to square one of our strategy -- identify the *attitudes* and *deal with them the best way possible.*

This paper discusses the attitudes towards using Ada at NSA, and after analyzing the negative attitudes, we present several methods for transforming people's view of Ada and promoting its adoption. Part of this includes discussing the progress and success of current and previous NSA Ada projects.

Attitudes Towards Ada

Initially, software developers within NSA resisted using Ada citing all the common reasons such as: "There are no compilers that run on our platforms."; "The compilers that exist are too slow and too expensive."; "We cannot get our people Ada training."; "It takes too long to learn the language."; and "We don't build embedded weapons systems." [MOG91] Even with the advent of more efficient and less expensive compilers, the availability of internal and external training courses, and the proof of Ada's capabilities for systems other than embedded weapons systems, the critics remain vigorously opposed to Ada. They have adopted new reasons such as: "All our systems are in C and UNIX; it would be too costly to rewrite them."; "There are no interfaces to commercial products." -- both valid but not insurmountable obstacles -- "Ada doesn't buy me anything new."; "Ada is too wordy."; and "All our people already know C." -- attitudinal problems. Some of these reasons are actual barriers; unfortunately, others are merely technical excuses masking negative attitudes towards Ada.

Why is it important to identify the attitudes and their affect on the acceptance of Ada? Since NSA is a technical-based Agency and has a significant in-house development effort, technical decisions affecting system performance are often delegated to the technical development team. The technical community does not just blindly follow the technical policies established and dictated by management, but also has an active voice in defining these policies. This community's overall attitude towards a new technology is a major factor in whether the technology is widely accepted and used. This overall attitude varies from endorsement to opposition. In the case of Ada, the predominant attitude is resistance.

People in the Intelligence Community, though often considered their own breed, are just as resistant to change as others. The assertion that Ada and software engineering are radical novelties [GRO90] certainly applies to NSA, since the adoption of these techniques requires significant changes in the current software development process. The perception that Ada has not proven itself and shows no significant advantage over other languages such as C or FORTRAN [MAR91] contributes to the lack of motivation by NSA's technical community to learn about or crossover to Ada.

The resistive attitudes do not only come from the technical community, but from management as well. Managers at NSA feel that developing systems in Ada, at this time, is on the "bleeding edge" versus the leading edge of technology. They consider Ada a risk area [GER90] -- unproven and immature -- and as Dr. Salwin observed, "decisions are most often based on perceptions, not reality, and decisions on Ada adoption are no exception." [MOG91]

Additionally, NSA links itself more closely to the commercial sector in technological areas because many of its systems are built on and use commercial-based hardware and software, rather than the special purpose systems traditionally built by others in DoD. Because of the nature of many of the systems, NSA also parallels the telecommunications world in related areas. Due to these strong ties, NSA's technical community also capitalizes on the strong resistance of the commercial industry toward Ada, as noted by Reed. [MOG91]

"The first problem for all of us, men and women, is not to learn, but to unlearn." [GRE84] Perhaps the largest contributing factor to the resistive attitudes within the technical community is due to their first experiences with the language. Many of the early reviewers and users of Ada experienced all the early problems of the language, leaving them with a negative feeling. Today many of those people are in management or senior technical positions. They are making decisions on which language to use, contributing their respected opinions to the technical world, and mentoring junior technical people. Because of this, the great improvements that Ada has made go virtually unnoticed. It's like the old saying goes, "A bad rumor spreads faster than a good one." Whenever Ada is mentioned, the perspectives of the early users prevail over the current state of the language.

Having gained an understanding of the resistance to Ada at the National Security Agency, we devised a plan to attack the barriers and transform people's negative views of Ada. The plan addresses the procurement and availability of adequate Ada development tools, education of software engineers and managers, resistance at NSA to changes and risks, and the commercial sector's support.

Promoting Ada

The task of promoting Ada in a resistive environment is difficult, but not impossible. The goal is to shift the technical workforce's attitude from resistance to open-mindedness towards Ada -- to evaluate Ada objectively.

Our methods for diminishing and removing the older, more common technical barriers primarily concentrate on supplying offices with development tools and training. This includes procuring, evaluating, and giving away compilers and tools for as many of the platforms as possible; establishing in-house and contracted training courses; and providing information on projects, other than embedded weapons systems, that are successfully using Ada.

Addressing the newer technical barriers is a bit more difficult because it relies heavily on promises of the future. Issues concerning interfaces to commercial products or the population of Ada-knowledgeable programmers require a look into the crystal ball. There is, however, a strong foundation. Many of the products NSA's technical community uses have or are developing Ada interfaces (i.e. X-Windows, Ingres, TeleUse, etc.) Unfortunately, many still do not. As for the knowledge base of Ada programmers, until the number of Ada-knowledgeable college graduates increases, the method of populating the Ada knowledge base remains in-house and contracted courses. Unfortunately, these reach only a minute number of our technical community.

Concerning NSA's long legacy of C and UNIX, this is not a trivial matter. Indeed, it would be very costly to rewrite all these systems; however, the purpose of using and promoting Ada is not to rewrite all these systems. Admittedly, there are existing non-Ada systems that either do not require much maintenance or were written with good engineering principles. The purpose is to use Ada for new development efforts and for those systems, or the parts of those systems, that require high maintenance activity -- whether enhancement, correction, or adaptation. The hope that the majority of systems will be partially or completely coded in Ada remains only a hope until the attitudes of the technical community are more receptive to Ada.

So the question becomes, "How do you teach the 'old dogs' new tricks?" When this question was posed to a distinguished panel of software engineering professionals, they answered, "Wait for them to die." [NAS90] Though said in jest, the response held a lot of truth, many 'young pups' view the situation similarly. We hope and are trying to prove that this is not necessary. By either retraining or perhaps just by outnumbering the 'old dogs', the 'young pups' can keep the team current on software engineering techniques and new technology.

"When you cease to make a contribution you begin to die." [GRE84] Encouraging the technical community to keep up with the current state-of-the-practice and to invest the time in their own education has also helped promote Ada. A large number of NSA's computer scientists and engineers work toward Masters degrees after hours, and suggesting that they take a software engineering course and an Ada course has helped build the knowledge base.

By educating the technical community, not only does it increase the knowledge base, but it also reduces the "Ada is such a difficult language to learn" syndrome. When approaching any new experience, including Ada, it is important to remember, "Nothing in life is to be feared. It is only to be understood." [GRE84]

The resistance of the technical community to adopting Ada also works to promote it. Using Ada at NSA is viewed more as a technical choice than as following a mandate. Pointing out that certain projects have investigated Ada and have chosen to use it for a specific application gives developers of similar systems another option. It also peaks the curiosity of managers and technical personnel who have never, or at least not lately, looked at Ada; as well as giving more backing to those who would like to use Ada but are discouraged from doing so. This also has an impact on the commercial world.

Because of the close ties NSA has with industry, its technological choices directly influence the commercial sector it deals with. Requesting compilers, tools, and interfaces to standard as well as non-standard platforms and products, advances the industry. Procuring and evaluating commercial products also promotes Ada's use outside the government arena.

Another part of the promotion process includes identifying the current and prospective Ada projects and determining how we could support them, as well as exploit their qualities.

Projects Using Ada

NSA's largest support for Ada comes from offices that have developed or are developing Ada projects. Several Ada projects currently underway vary in size, complexity, and purpose, and represent a good sample of the types of systems at NSA. By showing the progress and success of these projects and discussing some lessons learned, we hope to demonstrate the advantages and benefits of Ada.

Project 1

One project primarily built using the Ada programming language provides a community of workstation users with processing services which aid and support a special class of signal processing. The users of the system include analysts, managers, maintenance personnel, and remote users connected via their own host system. It provides an interface to other external systems for the purpose of storage and retrieval of working-aids, reference materials, and documents, as well as for exchange of electrical mail messages.

Roughly 75% of the total 500,000 source lines of code (SLOC) are written in Ada with the remaining SLOC being a variety of languages such as SQL, Pascal, FORTRAN, and VAX/VMS DCL. Due to its enormous size, the population and proficiency of Ada professionals has been increased. Also, this project has Ada interfaces to many commercial off-the-shelf (COTS) products such as Sybase and TCP/IP.

Currently in the software integration and test phase, this project will soon be transitioning to the test and evaluation phase. But already several key lessons have been learned from this project regarding the memory needs of Ada. For instance, Ada generates code to provide the default initialization of variables if they are not declared, consuming memory space and elaboration time. By declaring the initial values of variables and constants wherever possible, these problems can be reduced. In addition, string slicing was found to be inefficient both in terms of size and speed.

Project 2 and Project 3

Two other projects being developed in-house are subsystems to the previously discussed project. Both are working-aid systems. The first of these two projects is a multi-source, multi-media, multi-level secure database that includes a variety of application programs which access the database. The other project's development requirement is a set of target and topical working-aids. Since this project requires the exploration of special purpose technology to extend and enhance a widespread functional set, there is a high possibility of reuse on other workstation designs and systems.

Together these systems will contain approximately 200,000 SLOC of which 35% is being developed in Ada; the remaining 65% is SQL. Both projects use the VAXstation 3600 server, VAX 3100 workstations running VMS as the development platform, and associated environment tools; yet the target hardware will include Sun workstations running UNIX as well. Ada interfaces constructed or used include DECWindows and Sybase.

To reduce the risk of future Ada developments and serve as proving grounds, these efforts provided initial Ada project experience for software personnel and managers.

Project 4

Another Ada project accepts preformatted data from up to twenty input systems, and performs activity detection and signal formatting for forwarding to any of six output systems. Using the MV/Ada compiler and debugger with the Data General AOS/VS operating system on a Data General MV-series hardware platform, a defense contractor team has developed 129,000 Ada SLOC and an additional 14,000 SLOC in other languages.

The compiler was very new and had several problems. As the project progressed, bugs and flaws in the compiler surfaced. The defects were fixed under a joint effort between the development team and the compiler vendor. Now the compiler is more stable and usable. No other tools are being used.

Project 5

The final project's primary function is process control of hardware devices and operator interfaces. The system receives data from special purpose processing hardware over a fiber optic LAN; processes the data, which includes collecting it by reordering and segmenting it; then forwards the data via another fiber optic LAN to a follow-on system.

The estimated SLOC for this system is roughly 150,000, of which 130,000 is newly developed Ada code and 20,000 is reused Ada code. Only a minimal amount of SQL will be developed. The development environment includes IBM RS6000 workstations with a Rational R1000/300 co-processor and a Rational R1000/400 processor. Development tools being used include Software Through Pictures and FrameMaker integrated with the Rational environment, and VADS for the IBM workstation.

Following a software-first approach, the selection of a majority of the target hardware has been delayed until the critical design review. This minimizes risk by involving the software team early in the process and reduces the system constraints levied on them.

Currently, this project is approaching its Preliminary Design Review (PDR).

Small pockets within NSA are having success developing systems with Ada, although the real payoffs in maintenance have yet to be measured. Since this is a small number of the overall projects at NSA, it is important to broadcast each success to all our software developers to promote the benefits of using Ada.

Conclusion

The National Security Agency, like others in the Ada community, has barriers to the adoption of Ada. At NSA, the barriers are more attitudinal than technial and are very resistive. The attitudes derive from resistance due to change, resistance due to risk, little support from the commercial sector, preconceived ideas, and educational stagnation. To reduce and eliminate this resistance, we are promoting Ada by supplying tools, retraining and educating personnel, making technical choices instead of mandated decisions, and influencing the commercial industry. We are also promoting Ada by identifying and discussing Ada projects at NSA; proving that Ada can be used with little risk and that in certain application areas it has already proven itself.

Though the task of promoting Ada in a resistive environment is difficult, we have made progress. However, we realize there is still work to be done.

References

[GER90] Gerhardt, Mark S., "Sociological Concerns About Ada in the 1990s." Proceedings of TRI-Ada '90 Conference, Baltimore, MD, December 1990, p. 429.

[GRE84] Great "Quotes" From Great Women!, Great Quotations Inc., 1984, pp. [Madam Curie, W13], [Eleanor Roosevelt, W34], [Gloria Steinem, W7]

[GRO91] Gross, LtCol. R. and Maj. D. Umphress, "Software Engineering as a Radical Novelty: The Air Force Ada Experience.", Proceedings of TRI-Ada '90 Conference, Baltimore, MD, December 1990, pp. 501-507.

[MAR91] Marsh, Alton, "Is It Time to Throw Ada Out?", Government Executive, April 1991, pp. 39-40.

[MOG91] Mogilensky, Judah, "Summary of TRI-Ada '90 Panel Session: Barriers to Ada Adoption: Have They Changed?", ACM Ada Letters, Volume XI, Number 4, May/June 1991, pp. 12-17.

[NAS90] "Software Engineering in the 1980s: Most Significant Achievements/ Greatest Disappointments", (panel), Proceedings of the Fifteenth Annual Software Engineering Workshop, GSFC, Greenbelt, MD, November 1990.

About the Authors

Ms. Kimminau is the Ada Program Manager for the National Security Agency. Before moving into that position, she was a designer and developer on two in-house Ada projects. She received a Bachelor of Science degree in Computer Science from Colorado State University, a Master of Science degree in Computer Science from the Johns Hopkins University, and is currently working on a Doctor of Education degree at the George Washington University.

Mr. Graves is a lead software engineer in the Signals Processing Software Division at the National Security Agency. He has participated in two Ada development projects over the last five years and continues to promote the benefits of Ada. He received a Bachelor of Science degree in Computer Science from the University of Pittsburgh and a Master of Science degree in Computer Science from the Johns Hopkins University.