Special Panel
The First 100 Days of Ada

Moderator

Marianne E. Erdos
Grumman Data Systems

Panelists

Ralph Crafts, *SS&T*
Robert Dewar, *New York University*
Kenneth Fussichen, *Computer Science Corp.*
Robert Roe, *Boeing Corp.*
John P. Solomond, *Ada Joint Program Office*

PANEL THEME STATEMENT
IT'S THE LAW: THE FIRST 100 DAYS

Marianne E. Erdos
GRUMMAN DATA SYSTEMS

"Notwithstanding any other provision 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 any official designated by the Secretary of Defense."

What has the first hundred days of the mandated use of Ada brought to the software industry?

Historically, computer languages mature naturally in 10 to 15 years. For the Ada language the process was altogether different. On the one hand, the Ada language was developed under the tutelage of the DoD which led to the label: "the DoD mandated language." But the software community frowns upon mandating, even those involved with defense software acquisition and development. Energy has been spent to get waivers instead of getting accurate information about the validity of using Ada and software engineering. It still lingers on today. On the technical side, Ada was ahead of its time with its strong support of software engineering principles. These principles are just now coming into use and the tools to support software engineering are just now being placed into the hands of the practitioners. While "hacking" was acceptable for software development ten years ago, and it is still common for small, simple systems, today's large, complex systems require a formal engineering discipline.

Ada language has periodically been "encouraged" by DoD directives and various implementation policies by the Armed Services since its standardization in 1983. DoD directive 3405-1 specifies that "the Ada programming language shall be the single, common, computer programming language for Defense computer resources used in intelligence systems, for the command and control of military forces, or as an integral part of a weapon system." DoD directive 3405-2 establishes DoD policy and prescribes procedures for using the Ada programming language in computers integral to weapon systems. The availability of waivers, coupled with the procuring agency's lack of Ada knowledge and software engineering, led to a tolerance of alternative languages. The waiver approval process is becoming more stringent, however, the scarcity of metrics for software in general, and for Ada in particular, still weighs against the use of Ada.

This Panel will argue about the pros and cons of the part of the law that effects software programming, specifically, the mandated use of Ada. The law is in effect since June 1, 1991 and the legislation, entitled Section 8092 that includes the Ada language, is quoted above. Panel members represent different areas of expertise and cover the "Tri" - industry, academia, and government.

Biography

Marianne Erdos is a Systems Architect with Grumman Data Systems where she consults on Ada related software developments including methodology and tools. In her previous position, she was manager of the central Software Development Environment for embedded systems and established the Ada Resource Center. She attended here first Ada language course in 1982 which was conducted, as usual in those days, without hands-on experience. She is a member of SIGAda and established the Long Island SIGAda group in 1989.

THE FIRST 100 DAYS OF ADA MANDATE
(FY 91 DEFENSE APPROPRIATIONS ACT)

There has been much discussion so far about the impact of the "congressional mandate" on the overall health of the Ada industry. The intention of Ada Mandate was to encourage the use of Ada on all DoD systems, when previously it has been generally accepted policy to require its use on embedded or "mission critical" systems.

In the past, the DoD has implemented an increasingly strict policy relative to Ada waivers; the Air Force and Army policies, published in the summer of '90 reflect this trend. In the past, waivers have generally been granted on the basis of either technical or economic justification; now, under the new statute, the only criterion is an economic one, cost effectiveness--which DoD is implementing on the basis of life cycle costs.

While the number of Ada waiver requests have been declining over the past two years, I predict that decline will precipitate even more rapidly, to the point that the issue of waiver policy will eventually be less critical than it has been in the past.

Besides being concerned with the diversity of programming languages still being used within the DoD, Congress also recognized some of the explicit benefits of Software Engineering with Ada. The language of the House report specifically mentioned: "training economies due to standardization . . ., reduced costs with internal checking to inhibit errors . . ., separation of software module interfaces from their implementations in order to facilitate reuse", et al.

I am happy that Congress has shown a new awareness of the relevant technical issues associated with software technology. Furthermore, I am confident that the congressional mandate for Ada will benefit both the DoD and the commercial Ada community, as well as signaling to the rest of the world the US commitment to the language.

John P. Solomond
Director, Ada Joint Program Office

BIOGRAPHICAL SKETCH: JOHN P. SOLOMOND

John Solomond is the Director of the Ada Joint Program Office, he previously was the head of Computer Resource Management for Headquarters US Army Materiel Command. Furthermore, he has the additional responsibility as the NATO International Programme Manager for the Portable Common Interface Set (PCIS) Programme. This international effort involves the cooperation of ten NATO countries as well as the NATO Communications and Information Systems Agency (NACISA).

He is a member of the Institute of Electrical and Electronics Engineers (IEEE) and a referee for IEEE Software Magazine. He is also Chairman of the Federal Advisory Board known as the "Ada Board".

He received his B.S. and M.S. in electrical engineering from Carnegie-Mellon University, and his Ph.D. in operations research from Texas A&M University.

POSITION PAPER: IT'S THE LAW: THE FIRST 100 DAYS

Ralph Crafts
SOFTWARE STRATEGIES AND TACTICS, INC.

The Ada "law" developed by Congress in the FY91 Defense Appropriations bill is definitely having an effect. Although Desert Storm sidetracked many DoD initiatives, the Congressional deadline of June 1, 1991 caused last-minute scrambling throughout the DoD to comply with the mandate. However, the "law" is not a guarantee of compliance, nor is there a guarantee that the law itself will endure.

There are factions within the DoD that would like to see Ada go away. In the DoD's FY92 budget request, on pages 540-41, there was a recommendation that Section 8092 (the Ada "law") be deleted from the FY92 budget. Fortunately, Congressional staff members caught the "small print" recommendation, and kept the mandate in place in the FY92 House Appropriations report. As of the date of this position paper (June 18, 1991) the Ada law was alive and well in Congress, although the final budget vote will not take place until later this year.

The first military service to respond to the law was the Air Force. Their policy statement interpreted the words, "where cost effective," to mean that use of a language other than Ada must be proven more cost effective than Ada. Further, any up front Ada transition costs, including compilers, tools, environments, training, etc., must be amortized over the life of the current project and any future Ada projects which would use the tools and resources. This interpretation and policy is significant in many ways.

The treatment of Ada acquisition/transition costs as amortizable makes a great deal of sense if DoD is truly interested in making legitimate software "investments." For the acquisition of any other long-term resource, the costs would be amortized or depreciated over the resource's useful life; in other words, it would be treated as an asset. An investment in Ada tools, software engineering environments, etc., should also be treated as an asset, since it has long-term useful life. This treatment of Ada-related costs serves to remove the "Ada's too expensive" argument that has been used to justify Ada waivers.

Amortizing Ada transition costs should encourage more investment in Ada, and could lead to changes that will provide incentives to develop reusable software. If software that is explicitly designed for reuse can be carried on an organization's financial records as an asset, then corporate managers have a strong incentive to invest in reusable code. The major barriers to reuse have little to do with technology, however, rather, the legal and liability issues associated with the reuse of existing code provide far more difficult barriers to reuse than any of the technical challenges.

All of the potential benefits, incentives, etc., come down to interpretation and enforcement of the mandate. If poorly trained people are allowed to interpret and apply the Ada policies, there will be little if any, progress made in overcoming DoD software deficiencies. The bottom line, as always, will come down to human resources -- without a significant improvement in the capabilities and training of the people who specify, acquire, design, write, and support software, the Ada mandate/law/requirement will be for naught. By the time TRI-Ada '91 takes place, we should have some strong initial indicators as to how well DoD intends to address the real issues regarding Ada.

Biography

Ralph Crafts has been active in the Ada market for more than ten years. He is the founder and president of Software Strategies and Tactics, Inc. (SS&T), an Ada consulting and services company. He is also the Editor and Publisher of the *Ada Strategies* newsletter, and international publication dealing with Ada-related issues and trends. He serves as the President of the Ada Software Alliance (ASA), where he works with members of Congress and their staff on Ada and other software-related issues. He is a member of SIGAda, IEEE Computer Society, the IEEE's Committee on Public Policy (COPP), and the American Defense Preparedness Association. He was the Chairman of the first TRI-Ada Conference and Exposition in 1988, and has served in several roles for other Ada conferences and meetings.

It's The Law: The First Hundred Days

Kenneth Fussichen

Computer Sciences Corporation

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

This law elicits three separate and distinct views. Briefly they are:

- it is an indictment of the computing leadership of the Department of Defence that Congress has to force DoD to implement its own policies;
- it is ludicrous for Congress to mandate software engineers and computer scientists what computer language is or is not appropriate for any computer development for which they are not a direct client; and
- practically speaking, it doesn't make any difference.

As I recall, the entire Ada development process was put into place by a Department of Defence initiative that attempted to stem the costs of software maintenance. No language in existence at the time supported the software engineering principles necessary to cut maintenance costs. It was thought that a new programming language, focusing on those engineering principles could produce more reliable and maintainable code.

Now that the language exists in a reasonably robust form, and the early returns on the use of the language indicate that it does seem to be more maintainable and more reliable than previous computer languages, it takes an Act of Congress for DoD to implement its own policy.

Many of the arguments against Ada contrast Ada's slower development time to some favored language. It is unreasonable to compare Ada on terms for which it was not initially intended. Ada effectiveness cannot be judged solely during the development phase of the life cycle. Ada effectiveness can only be judged against the entire life cycle, to include maintenance.

The logic is as follows: In order to get the intended long term cost savings from Ada maintainability, it will be necessary to invest in the potentially higher cost Ada developments and move as rapidly as possible to the maintenance phase of the projects.

Restated, someone in DoD is going to have to bite the bullet. The squabbling over which language to use has to stop. It is not a matter of choice, it is a matter of spirit. The entire intent of the language was to reduce DoD maintenance costs. Ada has demonstrated herself to be eminently maintainable. It is time for DoD managers to quit belly-aching and sign up to the spirit of what was intended.

Ada was intended to be cost effective through the maintenance portion of the life cycle, not

just the development portions. Over time, Ada developments will likely become as quick as other languages for development purposes, perhaps even as quick as 4GL's. Now, the Ada advantage is clearly maintenance, which was the DoD intent was back in the late 1970's.

Part of the problem of slow Ada developments is the issue of reusability. Globally effective and practical reuse libraries have not been implemented. Reuse issues are stuck in a legal quagmire over who owns reusable code and how the costs of developing reusable code can be recovered. Reusable code costs more to develope, and currently there is no incentive for developers to create reusable code. These are complex issues that have implications to our current copyright, patent and trademark laws.

Given the large issues that require legal attention and expertise, I am at a loss as to the exact technical qualifications of the Senators and Congressmen who drafted this law and then voted it into existence. It's not that I don't appreciate their thoughtfulness in having removed the decision making burden from all those cackling and opinionated software engineers and computer scientists with regard to what computer language should be used, I just thought that all that cackling and all those opinions would ultimately lead to a correct decision based on a technical correctness appropriate to the particular client community. Apparently I was wrong.

All that said, the law is nothing more than a statement of intent. Other laws, DoD exceptions, "cost effective" conditions, and the noticeable lack of penalties for non-compliance, combine to provide adequate avenues to bypass any Ada requirement.

Bio. Ken Fussichen is a Computer Scientist with Computer Sciences Corporation. He served as a Project Manager and an architect for the STANFINS-Redesign project and also works with the Ada 9X Language Revision Project on IS-Ada issues. He can be reached at P.O. Box 16008, Indianapolis, IN 46216; voice: 317/543-6581; email: fussiche@ajpo.sei.cmu.edu.

Panelist Viewpoint

R. Mitchell VanDuyn

Mandates to use a particular language are not always effective means of achieving change. Such mandates are appealing because of their simplicity, and because they have the appearance of providing leadership and direction. On the other hand the measurable results of mandates can be quite small, or in fact counter productive.

In contrast to the US government mandate to use Ada, is the success of the US in the Persian gulf. I have observed that success in the gulf has raised the credibility of the Department of Defense (and hence by association Ada,) while the mandate has hurt Ada by making it appear that the language cannot survive on its own merits. In the end the effects of the mandate and the gulf conflict have balanced each other out.

One problem is that mandates tend to become emotional touch stones. For example a mandate to use Ada (and such mandates are not only made by the US government) often prompts the immediate reaction of: "If Ada is so great then why must its use be mandated" This reaction represents the whole problem that people dislike having solutions imposed on them, and if such solutions are imposed the tendency is towards rebellion, not towards acceptance. This has certainly been my experience within our company. On the other hand a positive emotional response occurs when people see success (where success is defined as meeting clear goals.) People are drawn to success stories and desire to emulate the methods of successful organizations and people.

The second problem with mandates is more fundamental: A mandate that dictates the use of Ada really has a hidden goal. What is being asked for is not actually use of Ada at all, but one or more of the following: Higher software development productivity; Lower software maintenance costs; Portable software; Reusable software. Not only does the simple "Ada mandate" not ask for these things, but worse it is quite possible to deliver Ada software (as per the stated requirement) without actually meeting any of the unstated requirements.

This does not mean that customers of software do not have the right to give specifications to their suppliers. It does mean that such specifications should be clearly measurable, and be clearly related to the customers problems and goals. In addition it is reasonable for the customer to aid in the improvement of the software development infrastructure. Mandates to use Ada cannot be met in a productive way until there is a competitive base of Ada vendors, an aggressive educational program, and evidence that Ada works. Effort spent on putting this infrastructure in place is much more effective than developing and enforcing mandates.

Mitch VanDuyn has been at Xerox Corporation since 1981. For the last two years he has been involved in the investigation of commericial programming languages including Ada for use in copier/duplicator products. Prior to this he worked both on product program development teams, and in a software tools development group.

These views are those of the panelist, and do not necessarily represent the position of the panelists employer.