ADA USER JOURNAL

Volume 25
Number 2
June 2004

Contents

Editorial Policy for *Ada User Journal*

Publication

Ada User Journal – The Journal for the international Ada Community – is published by Ada-Europe. It appears four times a year, on the last days of March, June, September and December. Copy date is the first of the month of publication.

Aims

Ada User Journal aims to inform readers of developments in the Ada programming language and its use, general Ada-related software engineering issues and Ada-related activities in Europe and other parts of the world. The language of the journal is English.

Although the title of the Journal refers to the Ada language, any related topics are welcome. In particular papers in any of the areas related to reliable software technologies.

The Journal publishes the following types of material:

- Refereed original articles on technical matters concerning Ada and related topics.
- News and miscellany of interest to the Ada community.
- Reprints of articles published elsewhere that deserve a wider audience.
- Commentaries on matters relating to Ada and software engineering.
- Announcements and reports of conferences and workshops.
- Reviews of publications in the field of software engineering.
- Announcements regarding standards concerning Ada.

Further details on our approach to these are given below.

Original Papers

Manuscripts should be submitted in accordance with the submission guidelines (below).

All original technical contributions are submitted to refereeing by at least two people. Names of referees will be kept confidential, but their comments will be relayed to the authors at the discretion of the Editor.

The first named author will receive a complimentary copy of the issue of the Journal in which their paper appears.

By submitting a manuscript, authors grant Ada-Europe an unlimited license to publish (and, if appropriate, republish) it, if and when the article is accepted for publication. We do not require that authors assign copyright to the Journal.

Unless the authors state explicitly otherwise, submission of an article is taken to imply that it represents original, unpublished work, not under consideration for publication elsewhere.

News and Product Announcements

Ada User Journal is one of the ways in which people find out what is going on in the Ada community. Since not all of our readers have access to resources such as the World Wide Web and Usenet, or have enough time to search through the information that can be found in those resources, we reprint or report on items that may be of interest to them.

Reprinted Articles

While original material is our first priority, we are willing to reprint (with the permission of the copyright holder) material previously submitted elsewhere if it is appropriate to give it a wider audience. This includes papers published in North America that are not easily available in Europe.
We have a reciprocal approach in granting permission for other publications to reprint papers originally published in *Ada User Journal.*

Commentaries

We publish commentaries on Ada and software engineering topics. These may represent the views either of individuals or of organisations. Such articles can be of any length – inclusion is at the discretion of the Editor.

Opinions expressed within the *Ada User Journal* do not necessarily represent the views of the Editor, Ada-Europe or its directors.

Announcements and Reports

We are happy to publicise and report on events that may be of interest to our readers.

Reviews

Inclusion of any review in the Journal is at the discretion of the Editor.
A reviewer will be selected by the Editor to review any book or other publication sent to us. We are also prepared to print reviews submitted from elsewhere at the discretion of the Editor.

Submission Guidelines

All material for publication should be sent to the Editor, preferably in electronic format. The Editor will only accept typed manuscripts by prior arrangement.
Prospective authors are encouraged to contact the Editor by email to determine the best format for submission. Contact details can be found near the front of each edition.
Example papers conforming to formatting requirements as well as some word processor templates are available from the editor. There is no limitation on the length of papers, though a paper longer than 10,000 words would be regarded as exceptional.

Editorial

I take pleasure in posting this editorial in the June issue of the Ada User Journal, administratively the first one in the 2004 cycle -- in which we will enjoy the company of larger set of sponsors --, for it marks our success in recovering a great deal of the very annoying delays incurred in the past production schedule. The new arrangement put in place at the beginning of the year seems to be working well: my thanks go to Santiago Urueña, our new News (☺!) editor and to Dirk Craeynest, our former News editor and holder of precious, long-standing experience with the Journal production, for their decisive collaboration in achieving this milestone. We will do our utmost to continue at this pace.

This issue of the Journal is especially rich with articles, in addition to featuring the usual wealth of News and Events. We do continue to follow very closely the proceedings of the Ada language revision process: we are grateful to Jim Moore, the convener of WG9, for offering a very comprehensive account of the shape and direction of this very important and very promising effort. Two more contributions proceed directly from the Tutorial programme of the Ada-Europe 2004 conference, held earlier this month in Palma de Mallorca: Peter Amey and Adrian Hilton of Praxis Critical Systems give us some flavour of the tutorial they offered on the production of high-integrity software; Mario Amado Alves, an active member of the Ada user community, provides an outline of his tutorial, which illustrates the Ada foundations for value semantics. As a means to extend the value offered to our readership, we have put in place a plan for reports and summaries from the Ada-Europe tutorial programme to systematically appear in the Journal. The two mentioned contributions inaugurate this new line. Finally, Louise Arkwright of ACT Europe reports on an Academic Program Initiative recently kicked off to extend the Ada awareness in the education arena. We do hope to be able to host more information on this initiative in the future!

Tullio Vardanega
Padova
June 2004
Email: tullio.vardanega@math.unipd.it

News

Santiago Urueña
Technical University of Madrid. Email suruena@datsi.fi.upm.es

Contents

Ada-related Events

[The announcements reported below are a selection of the many Ada-related events organized by local groups. If you are organizing such an event, feel free to inform us as soon as possible. If you attended one please consider writing a small report for the Journal. -- su]

Mar 18 - 11th Ada-Belgium General Assembly

From: Dirk Craeynest <dirk@heli.cs.kuleuven.ac.be>
Date: 16 Feb 2004 21:11:21 +0100
Organization: Ada-Belgium, c/o Dept. of Computer Science, K.U.Leuven
Subject: Ada 2005 presentation, Thu 18 Mar 2004 20:00, Ada-Belgium
Newsgroups: comp.lang.ada,fr.comp.lang.ada,be.comp.programming,nl.comp.programmeren

Ada-Belgium will hold its 11th annual General Assembly on Thursday, March 18, 2004, 19:00, at the U.L.B., Department of Computer Science, Boulevard du Triomphe / Triomflaan, B-1050 Brussels. The official convocation is distributed separately to members and is also available on the Ada-Belgium web-server.

At 20:00 the General Assembly will be followed by a technical presentation plus Q&A, by Pascal Leroy from IBM France.

[See also "Ada 2005 Presentation" in AUJ 25.1 (Mar 2004), pp.5-6 -- su]

From: Dirk Craeynest <dirk@heli.cs.kuleuven.ac.be>
Date: 4 May 2004 22:57:38 +0200
Organization: Ada-Belgium, c/o Dept. of Computer Science, K.U.Leuven
Subject: Ada 2005 presentation at Ada-Belgium event now on-line
Newsgroups: comp.lang.ada,fr.comp.lang.ada,be.comp.programming,nl.comp.programmeren

At the Ada-Belgium evening event earlier this year, Pascal Leroy, Senior Software Engineer with IBM France and chairman of the ISO Ada Rapporteur Group (ARG), gave a technical overview of the most important improvements that are currently under consideration for inclusion in Ada 2005.

We are pleased to announce that a copy of his presentation is now available on-line on the Ada-Belgium web pages.

Check out "What's new on the Ada-Belgium web-pages?" at URL <http://www.cs.kuleuven.ac.be/~dirk/ada-belgium/whatsnew.html> if you're interested.

From: Matthew Heaney <matthewjheaney@earthlink.net>
Date: Fri, 07 May 2004 01:39:27 GMT
Subject: Re: Ada 2005 presentation at Ada-Belgium event now on-line
Newsgroups: comp.lang.ada,fr.comp.lang.ada,be.comp.programming, nl.comp.programmeren

Note that Pascal's presentation includes a brief summary of the new standard container library.

For more detailed information see the latest release (2004-04-29 AI95-00302-03/03) of the AI-302 draft:

<http://www.ada-auth.org/cgi-bin/cvsweb.cgi/AIs/AI-20302.TXT?rev=1.5>

May 27 - Ada and Modeling

From: Laurent Pautet <pautet@inf.enst.fr>
Date: Fri, 30 Apr 2004 09:53:27
Subject: Journee thematique Ada et Modelisation
To: ada-france@ada-france.org

[Translated from French:] Ada-France is glad to announce a one-day thematic event on the subject of Ada and Modeling. The event will take place at Jussieu in room 203 (building 41) on 27 May 2004. The map of the location can be found at: www.upmc.fr/FR/info/Venir_UPMC/05

For logistic reasons, should you wish to participate, you should take contact with: Agusti Canals (agusti.canals@c-s.fr).

The event will include 9 presentations of 30 minutes each, followed by 15 minutes for discussion. The presentation will mostly focus on issues of applied modeling as well as on the latest news on Ada technology available to the user community.

The day event will be closed by the General Assembly of the association.

- EAST or the free software for Enhanced Ada SubseT
- Positioning Ada with respect to UML and MDA
- Modeling and Architectural Description with AADL
- HELIOS experience report on the use of Ada 95
- Modeling based on Ravenscar-oriented design patterns
- Modeling of concurrent applications using Petri Nets
- Modeling of distributed applications using CORBA
- Modeling of a CORBA ORB using Petri Nets
- Issues with the extraction of semantic properties using ASIS

Jun 14-18 - Ada-Europe 2004 Conference

From: Dirk Craeynest <dirk@heli.cs.kuleuven.ac.be>
Date: 21 Mar 2004 21:43:09
Organization: Ada-Europe, c/o Dept. of Computer Science, K.U.Leuven
Subject: 9th Int.Conf.on Reliable Software Technologies, Ada-Europe 2004
Newsgroups: comp.lang.ada,fr.comp.lang.ada

9th International Conference on Reliable Software Technologies - Ada-Europe 2004, 14 - 18 June 2004, Palma de Mallorca, Spain. http://www.ada-europe.org/conference2004.html

Organized, on behalf of Ada-Europe, by the University of the Balearic Islands, in cooperation with ACM SIGAda (approval pending) and Ada-Spain

Ada-Europe organizes annual international conferences since the early 80's. This is the 9th event in the Reliable Software Technologies series, previous ones being held at Montreux, Switzerland ('96), London, UK ('97), Uppsala, Sweden ('98), Santander, Spain ('99), Potsdam, Germany ('00), Leuven, Belgium ('01), Vienna, Austria ('02), Toulouse, France ('03).

The 12-page Advance Program brochure with full information is available on the conference web site; the AP contains the list of accepted papers, as well as a detailed description of the tutorials. Use the "Program" link at the top to either view or download the PDF version or contact the conference chair to request a printed copy of the brochure. [...]

Quick overview

- Mon 14 & Fri 18: tutorials
- Tue 15 - Thu 17: paper and vendor presentation sessions, exhibition

Program co-chairs

- Albert Llamosí, University of the Balearic Islands (UIB), Dept. of Mathematics and Computer Science, Spain, llamosi@uib.es
- Alfred Strohmeier, Swiss Fed. Inst. of Technology in Lausanne (EPFL), Software Engineering Lab, Switzerland, Alfred.Strohmeier@epfl.ch

Invited speakers

- S. Tucker Taft, SofCheck Inc., USA. "Fixing Software Before It Breaks: Using Static Analysis to Help Solve the Software Quality Quagmire"
- Martin Gogolla, University of Bremen, Germany. "Benefits and Problems of Formal Methods"
- Antoni Olivé, Universitat Politècnica de Catalunya, Spain. "On the Role of Conceptual Schemas in Information System Development"
- Steve Vinoski, IONA Technologies, USA. "Can Middleware Be Reliable?"

Special session

- Pascal Leroy, IBM France & ISO Ada Rapporteur Group. "Ada0Y: An Overview"

Tutorials (full day)

- "Developing a Web Server in Ada with AWS", Jean-Pierre Rosen
- "Practical Experiences of Safety and Security-Critical Technologies", Peter Amey & Rod Chapman
- "Developing Fault-Tolerant, Time-Critical Systems with AADL, UML, and Ada", Bruce Lewis & Ed Colbert
- "Real-Time Java for Ada Programmers", Ben Brosgol

Tutorials (half day)

- "Programming with the Charles Container Library", Matthew Heaney
- "Probabilistic Worst Case Execution Time Analysis", Guillem Bernat
- "No Pointers, Great Programs. How to Stay on the Value Semantics Side of the Ada Way", Mario Amado Alves
- "Requirements Analysis with Use Cases", Alfred Strohmeier

Papers

- 4 invited papers and 23 technical papers on Application Programming Interfaces, Critical Systems Modeling, Distributed Systems, Real-Time Systems, Reflection and XML, Scheduling, Static Analysis, and Testing
- authors from 10 countries: Austria, China, Czech Republic, France, Germany, India, Portugal, Spain, United Kingdom, and USA

Exhibition

- 8 exhibitors already committed: ACT Europe, Aonix, Green Hills, I-Logix, LDRA Software Technology, Praxis Critical Systems, RainCode, and TNI-Europe, others expressed interest
- vendor presentation tracks for exhibitors

Social evening events […]

Registration

- includes copy of proceedings, published by Springer-Verlag in Lecture Notes in Computer Science series (LNCS) and distributed at event
- includes coffee breaks, lunches, social events
- three day conference registration includes conference banquet
- early registration discount up to May 16, 2004
- additional discount for academia, Ada-Europe and ACM members

For more info, latest updates, or to get printed brochures, see the conference web site at: www.ada-europe.org/conference2004.html

Aug 27 - Workshop on Architecture Description Languages

Workshop on Architecture Description Languages (WADL04). August 27, 2004 - Toulouse France. http://www.laas.fr/FERIA/SVF/WADL04

WADL04 is a part of the IFIP WCC World Computer Congress

August, 22-27, 2004 Toulouse - http://www.laas.fr/wcc2004/

Workshop Outline

The workshop on Architecture Description Languages will be held as part of the WCC 2004 Conference in Toulouse, France, from 22 August to 27 August 2004. This one-day workshop will consist of a Tutorial(s) followed by a Technical Session. The Technical Session will describe significant research as well as innovative development and application. We solicit original papers describing state-of-the-art research and development in all areas of Architecture Description Language. In order to favorize a closer interaction between academic and industrial networking research communities, both academic research papers and industrial contributions are welcome.

Scope of the Workshop

The aim of an ADL (Architecture Description Language) is to formally describe software and hardware architectures. Usually, an ADL describes components, their interfaces, their structures, their interactions (structure of data flow and control flow) and the mappings to hardware systems. A major goal of such descriptions is to allow analysis with respect to several aspects like timing, safety, reliability, ... This workshop will provide a forum for practitioners and researchers to discuss recent development around ADLs.

Topics of interest include (but are not limited to):

- Components, Connectors, Composition
- Semantics, Formalization
- Verification, Simulation, Test
- Tools and Development Environments
- Standardization
- Industrial Projects

Deadlines:

Submission deadline: 15 March, 2004

Notification of acceptance: 30 April, 2004

Camera Ready paper: 30 May, 2004

For more details about the conference please refer to the WCC Conf. Web site: http://www.wcc2004.org.

Oct 6-7 - Automotive, Safety & Security und Ada Deutschland Tagung 2004

From: Peter Dencker <peter.dencker@aonix.de>
Date: Mon, 26 Apr 2004 12:59:47
Subject: Automotive_ Safety & Security und Ada Deutschland Tagung 2004, Tagung und CfP
To: Dirk Craeynest <Dirk.Craeynest@offis.be>

[Translated from German] Herewith we wish to announce the Workshop on "Automotive - Safety & Security 2004" to be held on 6. – 7. October 2004 <http://www.automotive2004.de/> to be followed by the Ada-Deutschland Days on 7. – 8. October <http://www.ada-deutschland.de> in Stuttgart, and also wish to invite you to either one of the events or both at your choice.

The announcement of the events was first published on May 22.

Both events are supported by the Working Group on Security of the Association on Informatics. […]

Nov 14-18 SIGAda 2004

From: Ricky E. Sward <ricky.sward@ix.netcom.com>
Date: 7 Feb 2004 11:50:19 -0800
Subject: CFP for SIGAda 2004
Newsgroups: comp.lang.ada

Call for Participation - SIGAda 2004

14-18 November 2004, Atlanta, Georgia, USA

Constructing highly reliable software is an engineering challenge that can now be met in many domains. The SIGAda 2004 conference focuses on how the application of software engineering methods, tools and languages interrelate and on how features in Ada affect the quality of the resulting software. Papers that analyze Ada with respect to these factors or in comparison to other languages are especially welcome. SIGAda 2004 gathers industry experts, educators, software engineers, and researchers interested in developing, analyzing, and certifying reliable, cost-effective software. Technical or theoretical papers as well as experience reports with a focus on Ada are solicited. A brief list of topics include safety and high integrity issues, real-time and embedded applications, Ada & software engineering education, Ada in other environments such as XML and .NET, Ada and other languages, metrics, standards, analysis, testing, validation, and quality assurance. For a more extensive list of topics visit the SIGAda 2004 web page.

Contributions are solicited in six categories: Technical articles, extended abstracts, experience reports, workshops, panels, and tutorials.

We openly welcome contributions from educators and students. Educator grants are available that include conference and tutorial registration fees. These fees will also be waived for student authors whose papers are accepted for publication. An Outstanding Student Paper Award will be given for the best student contribution to the conference, and the winner will receive $500 and an Xbox (TM) video game system. Technical articles or experience reports from students could focus on such projects as comparing applications implemented in other languages and then re-implemented in Ada, mixed language development of applications, how Ada is used with XML or .NET applications, and/or software engineering education experiences with Ada.

Keynote speakers include Pam Thompson, Director of Software Engineering at Lockheed Martin Aeronautics Corporation, and Watts Humphrey, Fellow and Research Scientist at the Software Engineering Institute. [...]

Ada and Education

Algorithms and Data Structures with Ada

From: Claude Kaiser <kaiser@cnam.fr>
Date: Sat, 7 Feb 2004 15:10:34 +0100
Subject: Re: liste et arbres
To: ada-france@ada-france.org

[Translated from French:]

> I am looking for information on the programming of linear lists and tree structures in Ada. Any information on this regard is welcome.
> BATLLE Thierry
> 3 Rue André Malraux
> 81990 LE SEQUESTRE
> thierrybatlle@ifrance.com
> http://thierrybatlle.fr.st

Take a look at the book by Christian Carrez and the material of his data structures classes at CNAM:

C. CARREZ Structures de données en Java, c++ et Ada 95 (Masson 1997)
http://deptinfo.cnam.fr/Enseignement/CycleA/SD/
[…] Fichiers/bibSDAda.tar.gz
[…] Fichiers/Demos_SD.zip

This information was already posted on the Ada-France list, where you should find more details.

Both the URL of the classes and the ISBN of the book can be found by searching the web site of Ada-France.

Therein, you can also take a look at the material prepared by Feneuille on Ada education.

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Mon, 9 Feb 2004 16:31:45 +0100
Organization: Adalog
Subject: Re: liste et arbres
To: <ada-france@ada-france.org>

> Take a look at the book by Christian Carrez and the material of his data structures classes at CNAM:
> C. CARREZ Structures de données en Java, c++ et Ada 95 (Masson 1997)

Also consider « Algorithmes et structures de données avec Ada, C++ et Java » by Abdelali Guerid, Pierre Breguet et Henri, published by Röthlisberger (the publisher for the Technical Universities of the Romandie.).

This book (and others) is included in the commented Ada bibliography: http://www.adalog.fr/biblio1.htm

Ada-related Resources

Network Resources

From: Preben Randhol <randhol@pvv.org>
Date: Tue, 17 Feb 2004 07:35:28
Subject: Re: ada+network
Newsgroups: comp.lang.ada

Kuba Malczak wrote:

> Hi I am new to Ada, could you tell me if in Ada I can write network oriented programms? If yes, could you give me any links to resources connected with it?

Sure. Look at: http://www.rfc1149.net/devel/adasockets

(sockets. GNAT also has socket library so you can see which you like best)

http://libre.act-europe.fr/aws/ (AWS is a complete framework to develop Web based applications.)

From: Tom Moran <tmoran@acm.org>
Date: Tue, 17 Feb 2004 01:51:16 GMT
Subject: Re: ada+network
Newsgroups: comp.lang.ada

Take a look at www.adaworld.com - Ada Projects - Ada Internet Projects for a variety of things. Also www.adapower.com/reuse/clawweb.html for a very simple web server. You might want to use the search at www.adaic.com if you have some specific needs.

Browsing Ada Source Code

From: Preben Randhol <randhol@pvv.org>
Date: Tue, 17 Feb 2004 11:10:04
Subject: Re: ada packages
Newsgroups: comp.lang.ada

Kuba Malczak wrote:

> Could you tell me if anywhere is something like package browser. I want to browse functions with comments about themn, now I must open some package file and look for function that I want, is any simpliest way ?

Some choices:

Adabrowse, http://home.tiscalinet.ch/t_wolf/tw/ada95/

gnathtml (in GNAT), http://libre.act-europe.fr/GNAT/

AdaDoc, http://adadoc.sourceforge.net/

GPS (IDE), http://libre.act-europe.fr/gps/

Writing Linux Kernel Modules in Ada

From: Jeff C <jcreem@yahoo.com>
Date: Thu, 25 Mar 2004 00:39:00 GMT
Subject: Re: Ada runtime and Linux Kernel Module
Newsgroups: comp.lang.ada

Rohan wrote:

> I am in the very early stages of looking at writing a Linux kernel module in Ada for a University project. I have looked around and it seems to be possible, if tricky.
> The Big Online Book of Linux Ada Programming

(http://www.vaxxine.com/pegasoft/homes/16.html) has some information, and I have a test module written based on the example code there.
Any module would need to avoid anything which used the ada runtime.
There are references to pragma no_run_time but I have found that as long as I stear clear of using anything that requres the runtime there is no problem even without the pragma.
However I would quite like to use exceptions and allocate veriables from the heap. The BOBOLAP states:
"If you have the time, you can always copy some of the standard Ada packages to a separate directory and compile them into your GNORT project, effectively creating your own small, custom run-time system."
I have had a play around trying to do this, but am somewhat stumbling in the dark. Does anybody have any pointers to how I go about this? Or indeed any example Ada linux module code!

Another option is to look at something like GNAT for RTLinux.

http://rtportal.upv.es/apps/rtl-gnat/

From: Luke A. Guest <laguest@abyss2.demon.co.uk>
Date: Wed, 24 Mar 2004 21:42:57 +0000
Subject: Re: Ada runtime and Linux Kernel Module
Newsgroups: comp.lang.ada

The only thing you need is the system.ads file. The rest you don't need and won't really be able to use for OS level programming.

Another feature you *will* require is the use of "pragma Restrictions(No_Elaboration_Code);" to force the compiler not to generate things like functions for arrays, i.e. conversion functions for indexing.

Also, use the -gnatdg switch to see what GNAT is generating and you'll see what I mean.

As I remember it, GNORT doesn't exist anymore.

I also created a gnat.adc file with the following inside it:

```
pragma No_Run_Time;
pragma Restrictions
 (No_Exceptions);
```

That should be all you need.

From: Stephane Carrez <stcarrez@nerim.fr>
Date: Thu, 25 Mar 2004 21:06:13 +0100
Organization: Nerim -- xDSL Internet Provider
Subject: Re: Ada runtime and Linux Kernel Module
Newsgroups: comp.lang.ada

Have a look at MaRTE OS. They have a minimal GNAT runtime. There are probably a lot of ideas to use from their OS design (entirely in Ada).

http://marte.unican.es/

Ada OpengGL Mailing List

From: Chip Richards <spam@too.bad>
Date: 9 Apr 2004 08:06:57 GMT
Subject: [Announce] Ada OpenGL mailing list re-started
Newsgroups: comp.lang.ada,comp.graphics.api.opengl

Several years ago, we hosted a mailing list whose focus was the use of the OpenGL 3D graphics API from Ada code. Technical problems forced that list off the air, but we have established a new list, and hope to get some discussion going again. The new list's subscription address is: http://www.niestu.com/cgi-bin/mailman/listinfo/oglada

Note that unlike many OpenGL discussions, this one is not focused on game developers, though we do not exclude them. Our initial topics:

* First and foremost, any technical discussion of using OpenGL with Ada is welcome.
* There currently exist several distinct and partially incompatible thin Ada bindings for OpenGL. We would like to get those various groups talking, and see if it's possible to unify at least the lowest level. It would seem that a consistent interface to this standard API would be a Good Thing.
* All of the current bindings have an unmistakeable C flavor. While this is probably unavoidable, and is actually beneficial for porting existing OpenGL apps into Ada, it's less than ideal for (some) new development, where one might wish to work in a more purely Ada environment. We would like to develop a more Ada-aware interface, presumably "thicker" without sacrificing performance, but the design of such an interface is not trivial, nor particularly obvious. Come help us design bindings *you* would like to use.

We invite everyone with an interest in these topics to join the list to read and possibly contribute. Please disseminate this announcement to anyone who might be interested. And if this is a duplication of some existing effort, let us know.

Ada Concurrency

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Thu, 1 Apr 2004 19:07:26 +0200
Organization: Adalog
Subject: Re: distributed objects
To: ada-france@ada-france.org

[Translated from French :] Thomas De Contes wrote:

> http://153.103-30-212.9massy1-1-ro-as-i2-1.9tel.net/~thomas/adadistilled.pdf
That’s exactly what I needed to start with tasking :-)))
I would like 2 more things, if you allow me:-)
1 – Does anybody now where could I download it (I have received it by email)?

http://www.adaic.org/docs/distilled/adadistilled.pdf

The reference was (of course :-) on http://www.adalog.fr/adaweb.htm

From: Thomas De Contes <d.l.tDeContes@free.fr>
Date: Fri, 9 Apr 2004 04:58:06 +0200
Subject: Re: distributed objects
To: ada-france@ada-france.org

[Translated from French] Thomas De Contes wrote:

> 2-
"There are two models for Ada concurrency: multitasking, and distributed objects. The latter, distributed objects, is beyond the scope of this book. We focus this discussion on multitasking."
Does anybody know whether anything like that exists also for "distributed objects", please?

http://libre.act-europe.fr/Software_Matters/dl-distributed.pdf

The class material by Franco Gasperoni is truly excellent :-)))

We could as well use them with people that are not versed in informatics!

Smart Pointers in Ada

From: Preben Randhol <randhol@pvv.org>
Date: Sat, 28 Feb 2004 13:14:44
Subject: Re: smart pointers
Newsgroups: comp.lang.ada

Jorge Suárez-Solis Rivaya wrote:

> C++ smart pointers are equivalent to Ada smart pointers?

Ada smart pointers: http://www.adapower.com/alg/smartp.html

see also: http://home.earthlink.net/~matthewjheaney/charles/

Physical Units Checking in Ada

From: Christoph Karl Walter Grein <AdaMagica@web.de>
Date: Mon, 10 May 2004 10:46:13 +0200
Subject: Physical Units Checking
Newsgroups: comp.lang.ada

I've uploaded my paper presented at Ada Europe 2003 in Toulouse to my homepage: http://home.t-online.de/home/christusch.grein/Ada/Dimension.html

You can find there the comparison of four (five) methods how to deal with physical types in Ada.

From: Jacob Sparre Andersen <sparre@nbi.dk>
Date: 09 Apr 2004 17:48:58 +0200
Subject: Expressing physical units
Newsgroups: comp.lang.ada

The type system is not _unable_ to express physical units. It just has some limitations in how you can do it. And al-though I am annoyed by the limitations, I still haven't seen a description of how the language can make it easier, without introducing problems outweighing the benefits [...].

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Sat, 07 Feb 2004 11:29:59 +0100
Subject: Units of measurement for Ada v1.4
Newsgroups: comp.lang.ada

Bug fix: the Get procedures will raise Constraint_Error on numeric errors even if Machine_Overflows is not true.

www.dmitry-kazakov.de/ada/units.htm

From: Preben Randhol <randhol@pvv.org>
Date: Tue, 10 Feb 2004 15:43:33
Subject: Re: Dimension checking in Ada - impossible? In C++ it's possible using templates.
Newsgroups: comp.lang.ada

Preben Randhol wrote:

> Dmytry Lavrov wrote:

>> It's possible to do something similar in Ada? Have no idea how.

and

http://home.t-online.de/home/Christ-Usch.Grein/Ada/Dimension.html

Ada-related Tools

Simple Components v1.4

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Sun, 22 Feb 2004 14:01:14 +0100
Subject: Simple components v1.4
Newsgroups: comp.lang.ada

http://www.dmitry-kazakov.de/ada/components.htm

Changes:

1. The packages rooted in Parsers provide infix expressions syntax analyzers. Parsers can be used for syntax analysis of infix expressions, i.e. ones containing infix (dyadic), prefix and postfix operators, brackets, function calls, array indices etc. The approach presented does not require any grammar put down to generate scanner and analyzer. Nor any code generation steps are required. An object-oriented approach is used instead. The lexical procedures are dispatching, so that implementations may be provided through overriding them. Parsers can be used both for immediate one-pass code interpretation and for parsing tree building. Parser automatically detects the expression end allowing its easy integration. Operator precedence is expressed in a native way by setting priorities controlling association with the operands. Associations with the left and right side operands are controlled independently. Commutative operators and their inverses can be optimized when necessary. Especial attention is paid to error handling allowing generating very precise error messages and source code references. Samples from a small console calculator to a complete parsing tree generator for Ada 95 expressions illustrate examples of use;

2. The package Generic_Segmented_Stack provides a specialization of the generic stack package;

3. Random access to generic stacks (Get / Put);

4. Strings edit facilities;

5. Tables management package.

Charles - Container Library

From: Matthew Heaney <mheaney@on2.com>
Date: 10 Mar 2004 11:10:01 -0800
Subject: Re: Lists within Lists or Nested Linked Lists...
Newsgroups: comp.lang.ada

Charles is moving to tigris.org. The new URL is: <http://charles.tigris.org/>

I have just recently posted the latest release of the doubly-linked lists. The singly-linked lists will be done in a day or two.

From: Matthew Heaney <mheaney@on2.com>
Date: 15 Mar 2004 07:58:33 -0800
Subject: Re: Lists within Lists or Nested Linked Lists...
Newsgroups: comp.lang.ada

I have posted the singly- and doubly-linked list containers. There are both bounded and unbounded forms for each.

<http://charles.tigris.org/source/browse/charles/src/>

PragmARC - PragmAda Reusable Components

From: PragmAda Software Engineering <pragmada@earthlink.net>
Date: Sun, 21 Mar 2004 23:12:37 GMT
Subject: New Release of the PragmAda Reusable Components
Newsgroups: comp.lang.ada

PragmAda Software Engineering is proud to announce a new release of the PragmAda Reusable Components. This release includes Image functions for integer types that allow the caller to specify the width, base, and zero-filling of the image.

The PragmAda Reusable Components are located at

http://home.earthlink.net/~jrcarter010/pragmarc.htm

The mirror at www.adapower.com has not been updated for some time, and should not be used.

SAL 1.60

From: Stephen Leake <stephen_leake@acm.org>
Date: 05 Apr 2004 21:36:03 -0400
Subject: sal 1.60 released
Newsgroups: comp.lang.ada

SAL 1.60 is now released.

One major change is an implementation of Grace config_files is now in SAL.

SAL (Stephe's Ada Library, or Standard Ada Library :) is a collection of packages providing basic data structures; lists, stacks, binary trees, dynamic arrays, etc. Each package is a generic, providing maximum flexibility in the types that may be used with the package. There are both tagged (polymorphic/dispatching) and non-tagged data types, so you can use the one that fits your application best.

There is also a complete set of packages for spacecraft or robotics math. And lots of other stuff :). http://www.toadmail.com/~ada_wizard/ada/sal.html for more info.

Ada for Embedded Platforms

From: Jerry Petrey <jdpetrey@raytheon.com>
Date: Fri, 05 Mar 2004 16:54:39 -0700
Organization: Raytheon Company
Subject: Re: Embedded Tools
Newsgroups: comp.lang.ada

Carroll-Tech wrote:

> Well, I bought some robotics books (on sale for 1.00$). In it they use a 16f84 Microcontroller. That is an "embedded" technology right?
Then it dawned on me that I've seen many, many job openings for embedded programmers using Ada. And that tells me that Ada is the "native" language for "some" processor and that there is a compiler and libraries/package specs and implementation that might help me answer other questions.
So, I guess, for comparative reasons and having the books I bought the target is a 16f84 microcontroller. I just want to look in the specs and implementation of the libraries to program embedded "things" with Ada.

Ada is a high-level language. It is not the 'native language' (aka machine language) for any particular processor. There has to be a compiler that translates the high level Ada statements into the machine code of a given processor. Unfortunately, there is not such a compiler for most of the common 8-bit and 16-bit microprocessors or microcontrollers such as the PIC 16F84. There are compilers for processors such as the 80486, 60040, PowerPC, etc. and host compilers for Windows machines. It would be nice if there were more cross-compilers (one that runs on a machine like a PC and generates code for some other processor) for some of these smaller microcontrollers but so far that is not the case.

From: Bernd Trog <berndtrog@yahoo.com>
Date: 6 Mar 2004 11:47:50 -0800
Subject: Re: Embedded Tools
Newsgroups: comp.lang.ada

Try Atmel's AT90S2313. (2kByte Flash, 128 Byte SRAM in a DIL-20 package)

You can download a patch for gcc-3.3.2 here: http://avr-ada.sourceforge.net/

If you are new to Ada, I'd suggest to try compiling some small host-apps first.

Crosscompiler for PocketPC

From: Wojtek Narczynski <wojtek@power.com.pl>
Date: 6 May 2004 15:13:28 -0700
Subject: Re: Pocket PC
Newsgroups: comp.lang.ada

> I am trying to find any relevant info for developing for PocketPC 2002 and other similar devices in Ada, I have found a few odds and ends for embedded control systems, but nothing that relates to the consumer handheld devices.

We've been able to build GNAT cross-compiler for Sharp Zaurus, tasking and everything was working (passing most ACATS). Zaurus is a nice Linux based handheld on ARM / XScale. http://www.sharpusa.com/zaurus/

We never used it for anything commercial, though.

16-bit Ada Compiler for MS-DOS

From: Jeffrey Carter <jrcarter@acm.org>
Date: Sat, 15 May 2004 01:43:46 GMT
Subject: Re: 16-bit Ada for MS-DOS?
Newsgroups: comp.lang.ada

Victor B. Putz wrote:

> The platform in use is based on a Nec V53 chip running a proprietary OS, so the development environment is 16-bit C (or C++), compiled in small memory model using MS Visual C++ version "Old" (the command-line compiler is version 8.00c, from 1993 or so), and linked with several small-model libraries.
> I'm not sure I could even get the change approved if there WAS a solution, but is there a way to compile Ada for 16-bit C and linkable with MS small-memory-model C libraries? I've looked a bit at gnat and GCC, but I can't see how to specify a target like that...

S. Tucker Taft's company, SofCheck, has an Ada-to-ANSI-C compiler that should produce C that your C compiler could process. STT's e-mail address is stt-at-sofcheck-dot-com. (I haven't noted him posting here recently, so I've obsured the address.)

RR Software has a 16-bit DOS compiler available, if that will run on your target. Randy Brukardt is the contact there (randy@rrsoftware.com).

Compiler compatible with Borland 4.51

From: Vinzent 'Gadget' Hoefler <nntp-2004-05@t-domaingrabbing.de>
Date: Fri, 07 May 2004 15:06:23 +0200
Subject: Re: Object format
Newsgroups: comp.lang.ada

Henrik Quintel wrote:

> But I am forced to make use of the Borland 4.51 DOS Compiler which produces DOS Executables and DOS object files respectively (16 bit).

Bad luck. GNAT is a 32-bit-compiler, even under DOS. The object format *must* be incompatible. But perhaps you can use djgpp instead of Borland to compile the C(++)-files? djgpp is also based on gcc (BTW, djgpp should be able to compile Ada these days, too), so chances are that ez2load and djgpp have the same (or at least an easily convertible) object file format.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Wed, 12 May 2004 15:11:44 -0500
Subject: Re: Object format
Newsgroups: comp.lang.ada

The 16-bit version of Janus/Ada supported Borland C compilers back in the day. (It's still on one of my machines, because F-Prot thinks it's a virus! :-)

We still sell the 16-bit Janus/Ada compilers (I believe you'd need the professional version for this task). They host on DOS or Windows 95/987/ME. (The DOS Extender doesn't work on NT/2K/XP, unfortunately.) See www.rrsoftware.com or drop me a line.

Compiler for OS/2

From: Paul <pcas1986@bigpond.net.au>
Date: Sat, 21 Feb 2004 04:01:07 GMT
Organization: BigPond Internet Services
Subject: Ada & OS/2
Newsgroups: comp.lang.ada

Does anyone know what happened to the Alsys OS/2 Ada compiler (& toolset) after Alsys folded? I know the Win NT & Unix versions got picked up by Aonix but I have never seen any mention of the OS/2 version on their website. I wrote several Ada programs for OS/2 that are still in use but the current version of OS/2 (V4.5 convenience pack) crashes when running any of these programs. I dug out a copy of the compiler and it will not run at all - not the install program, the compiler or any of my old ada executables - not even the "Hello world" one liner. Obviously there is a major mismatch somewhere - perhaps in the system dlls.

Ok - I know OS/2 is dead but some are still running it and my client is one.

Our copies of the compiler are V5.5 circa 1995. I had heard there was a later version 5.6 maybe.

Is anyone still using this compiler? Was there a later version? TIA

From: Ed Falis <falis@verizon.net>
Date: Sat, 21 Feb 2004 14:26:24 GMT
Subject: Re: Ada & OS/2
Newsgroups: comp.lang.ada

Aonix still owns the rights to the product, but I'd guess they'd be hard-pressed to rebuild one. You might try contacting them, or try out the old (but much more recent) version of GNAT that you were pointed to.

From: Kees de Lezenne Coulander <keesl@adse.nl>
Date: Sun, 22 Feb 2004 20:03:23
Subject: Re: Ada & OS/2
Newsgroups: comp.lang.ada

I had never heard of an OS/2 version of the Alsys Ada compiler before, so I cannot help you in that line of investigation. In general, older OS/2 programs run very well on later incarnations of the operating system. The Convenience packs (v. 4.5) incorporate major changes to the kernel, but I would hesitate to blame that without further evidence.

Ada on OS/2 is not dead, though. Gnat on Intel PC started out as an OS/2 program, before Windows NT became the major PC operating system. Although ACT no longer provides ready-made binaries for the OS/2 version of the compiler, newer versions (up to the current 3.15p) are available thanks to the efforts of Dave Parsons.

I use gnat 3.14p on OS/2 (v. 4.0) very regularly on fairly large programs and have never had a problem with it. I have so far not moved up to 3.15p (sorry, Dave), but I hope to do that one of these days.

Gnat is of course Ada 95, whereas Alsys was Ada 83. That in itself should present no great problems. There are just a few minor issues which can be easily fixed. But if any Alsys-specific libraries are involved, that would of course be a different matter.

From: Georg Bauhaus <sb463ba@l1-hrz.uni-duisburg.de>
Date: Sun, 22 Feb 2004 00:28:44

Subject: Re: Ada & OS/2
Newsgroups: comp.lang.ada

There is some material at
http://www.unixos2.org/ada/

From: Dave Parsons <dwparsons@surfeu.de>
Date: Sun, 22 Feb 2004 07:14:52 +0100
Organization: CDL
Subject: Re: Ada & OS/2
Newsgroups: comp.lang.ada

or maybe:
ftp://ftp.cs.nyu.edu/pub/gnat/3.15p/contrib/os2/gnat-3.15p-os2-bin-20021124.zip

Compiler for OpenVMS

From: Britt Snodgrass <britt@acm.org>
Date: 11 Feb 2004 07:06:14 -0800
Subject: Re: Gnat for OpenVMS
Newsgroups: comp.lang.ada

Keith Brown wrote:

> Is Gnat for OpenVMS free like Linux or is there a license fee?

A very old GNAT version 3.12p is available at FTP address:
cs.nyu.edu/pub/gnat/private/old/openvms/

A.C.T. never provided newer public versions. If you need it for personal use DEC....Compaq.....Hewlet-Packard may be willing to provide GNAT as part of the VMS hobbyist license program (see: http://www.openvmshobbyist.com/hobbyist_faq/index.html).

KDevelop 3.0

From: Keith Brown <kbrown2720@comcast.net>
Date: Tue, 03 Feb 2004 19:50:17 -0600
Subject: KDevelop 3.0 released
Newsgroups: comp.lang.ada

The Kdevelop 3.0 programming environment was released today. This release includes support for Ada95 among other languages. This looks like a very exciting development to me (no pun intended). Is anyone in this group doing any Ada95 work (or planning to) with KDevelop? This appears to support an IDE and GUI development tools for the KDE environment. For a list of features see the link below. Any comments?

http://www.kdevelop.org/index.html?filename=features.html

Umbrello UML Modeller

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Wed, 04 Feb 2004 15:03:39 +0100
Subject: Re: KDevelop 3.0 released
Newsgroups: comp.lang.ada

A new Version of "Umbrello UML Modeller" was released as well:
http://uml.sourceforge.net/

Umbrello features an UML to Ada code generator.

[Umbrello UML Modeller is a Unified Modelling Language diagram programme for KDE. UML allows you to create diagrams of software and other systems in a standard format. Umbrello 1.2 is the result of a hard year's work by a dedicated team of developers. New features include improved code generation in more languages (Ada, AS, C++, IDL, Java, JavaScript, Perl, PHP, Python, SQL, XML Schema), undo/redo, a resizeable and zoomable canvas and more diagram types. It is part of KDE 3.2 -- su]

From: Oliver Kellogg <okellogg@freenet.de>
Date: 4 Feb 2004 23:08:08 -0800
Subject: Re: KDevelop 3.0 released
Newsgroups: comp.lang.ada

I will add Ada code import soon, but in order to use it people will need to compile umbrello from CVS or wait for version 1.3 (which won't be released for quite a while.) I am looking for people to help out with integrating the parser and code generator for true round-trip engineering.

Beware that the Ada support is really still in development, as can be seen in:
http://developer.kde.org/documentation/library/cvs-api/kdevelop/html/LangSupportStatus.html

Also, the kdevelop team is looking for active contributors/ maintainers of the AdaSupportPart:
http://developer.kde.org/documentation/library/cvs-api/kdevelop/html/unmaintained.html

Help from the Ada community is much appreciated.

GDS Compiler

From: Preben Randhol <randhol@pvv.org>
Date: Wed, 10 Mar 2004 10:10:50 +0000
Subject: GDS Compiler written in Ada
Newsgroups: comp.lang.ada

I was notified from freshmeat.net today about a new project in Ada.

GDS Compiler by Gilles Depeyrot

Aimed at micro-electronic design, GDS Compiler allows you to easily develop any GDSII generators (memory, MEMS, devices) of any macrocell for which parametrization can provide flexibility, security, and productivity. Front end views and netlist for LVS are generated automatically.

Don't know if it is of interest to anybody here, but you can have a look.
http://freshmeat.net/projects/gdscompiler/

Fast Fourier Transform in Ada

From: Dr Steve Sangwine <sjs@essex.ac.uk>
Date: Fri, 13 Feb 2004 12:36:05 +0000
Subject: Re: fast FFT code
Newsgroups: comp.lang.ada

Tom Moran wrote:

> Can someone point me to an Ada fast FFT code somewhere? In particular for 2D real correlations. I have the "fft_pack" (Glassman's algorithm) but it's quite general and can presumably be improved on. djbfft 0.76 claims to be the fastest around, but it's set up for Unix systems and I find it unreadable, and thus untranslatable, C.

Does the FFT have to be written in Ada? How about fast C code and an Ada binding? If this would do, look at FFTW (www.fftw.org) and my FFTW_Ada binding
http://privatewww.essex.ac.uk/~sjs/fftw_ada/fftwa.html
(not yet modified for the most recent release of FFTW, but it works with the previous one).

Multiprecision Numbers

From: Gautier
Date: Sat, 21 Feb 2004 11:51:07 +0100
Subject: Re: looking for a library to handle big numbers
Newsgroups: comp.lang.ada

Evangelista Sami:

> I have to write a program which manipulates big numbers. I am looking for a library which could do this. I have found some on the web but I don't know which one to choose. so if anyone has ever used one and give some hints, that would be very helpful.

At least there is a site about the various big number implementations for or in Ada: http://www.chez.com/bignumber/

NB: there is a newer version of my code in mathpaqs.zip, URL below (the files are: mupr*.ad*, test_int.adb, test_rsa.adb).

www.mysunrise.ch/users/gdm/gsoft.htm

AdaImgSvr 0.5

From: Patrice Freydiere <frett27@free.fr>
Date: Mon, 12 Apr 2004 16:34:01 +0200
Subject: AdaImgSvr version 0.5 released.
Newsgroups: comp.lang.ada

We are proud to annonce the version 0.5 of AdaImgSvr this new release implements :

- MySQL database support
- Win32 service integration
- Linux deamon utilities
- Security fixes.

http://adaimgsvr.sourceforge.net

Enjoy, and feel free for feedbacks

TeXCAD 4.1

From: Gautier
Date: Tue, 16 Mar 2004 22:50:00 +0100

Subject: TeXCAD 4.1
Newsgroups: comp.lang.ada

TeXCAD is a program for drawing or re-touching {picture}s in LaTeX. More info @ http://www.mysunrise.ch/users/gdm/texcad.htm

I'm still looking for people interested in developing the GUI part for other than "native" Windows:

- GtkAda
- "native" Mac OS X
- ?

A large part of TC is OS & GUI-independent (all TC.* packages), so it is "only" a question of windows, menus, mouse etc.

New Thick Binding for OpenGL

From: Luke A. Guest <laguest@abyss2.demon.co.uk>
Date: Sat, 13 Mar 2004 16:40:30 +0000
Subject: Anyone interested in Ada & OpenGL
Newsgroups: comp.lang.ada

Can come to irc.freenode.net #Ada.

I've been playing with the original work by David Holm, and have started to create a thick binding.

I'm usually there.

Library for BMP/JPG/GIF

From: Gautier
Date: Mon, 16 Feb 2004 23:59:42 +0100
Subject: Re: BMP/JPG/GIF library in Ada?
Newsgroups: comp.lang.ada
http://www.mysunrise.ch/users/gdm/gsoft.htm

Brian Catlin:

> Does anyone have a library for reading, writing and manipulating image files in the BMP, JPG, or GIF format? I tried searching on AdaPower, but the search capability doesn't work (nor do a lot of the links)

There are BMP (I/O) and GIF (I) in SVGA.IO in dos_paqs.zip, URL at bottom. Full Ada, easy to adapt to another contex as SVGA & DOS.

BTW, for PNG there is: http://privatewww.essex.ac.uk/~sjs/png_io/png_io.html

Texture mapping with Ada

From: Joachim Schröer <joachim.schroeer@web.de>
Date: Sat, 13 Mar 2004 15:00:38 +0100
Subject: Re: Texture mapping with Ada
Newsgroups: comp.lang.ada

Alfredo Macias wrote;

> Hi,
> Has anyone done texture mapping in Ada with openGL?

Look into www.adapower.com/schroer

Download ogl-src.zip

Look into opengl-earth.adb

There a bitmap of the earth is mapped onto a sphere with 2D texturing.

AdaMagick - ImageMagick Bindings

From: Ali Bendriss <ali.bendriss@chups.jussieu.fr>
Date: Fri, 12 Mar 2004 10:04:13
Subject: AdaMagick
To: ada-france@ada-france.org

[Translated from French] I have just posted on-line the first « release proposal » of a binding towards the C API of ImageMagick.

The binding is far from finished, to make it an interesting library it especially lacks the procedures for handling the BLOB (binary large object) data embedded in ImageMagick.

The object is not that of implementing all the functions in ImageMagick, rather to use ImageMagick as an interface to the most used image formats.

URL: http://karakush.free.fr/index.html

Please, have a look and let me have your feedback.

Any observation is welcome; I would expecially value comments on:

* types and constraints
* memory deallocation
* exception handling.

Thanks in advance.

Gwindows_Extended - Extension Packages for Gwindows Binding

From: Frank Piron <frank@konad.net>
Date: Mon, 19 Apr 2004 14:12:31 +0200
Subject: Gwindows_Extended
Newsgroups: comp.lang.ada

Gwindows_Extended contains packages which enrich David Botton's Gwindows Binding with state-of-the-art controls.

Gwindows_Extended is (of course) released under the GPL.

Some Features of Gwindows_Extended:

- Extended list view controls with header drag and drop, column sorting, coloring at subitem lev Icons at item level, Grid appearance and more.
- Extended toolbars with tooltips, flat buttons, check-style buttons and icon support.
- Extended tree view controls with icons and colors and a new event which occurs when node selection changes.
- A splitbar implementation.

Download from: http://www.konad.de/download.htm

Florist - POSIX Ada Binding

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Tue, 06 Apr 2004 10:50:57 +0200
Subject: Florist for GNAT 20040403 release
Newsgroups: comp.lang.ada

First snapshot release from Florist for GNAT.

Florist does not need original compiler sources.

Binary package for i686, GCC 3.5, SuSE 9.0 available. [URL: http://sourceforge.net/projects/gnat-florist--su]

mailto://krischik@users.sourceforge.net

http://www.ada.krischik.com

wxWindows/wxWidgets Binding

From: Lionel Draghi <Lionel.Draghi@Ada-France.org>
Date: Tue, 24 Feb 2004 23:32:58 +0100
Subject: Re: wxWidgets (wxWindows)
Newsgroups: comp.lang.ada

Szymon Guz wrote:

> Do you know if anybody created a binding to wxWindows in Ada?

Craig Carey: http://www.ijs.co.nz/code/ada95_wxwindows_cpp_bindings_unfinished.zip

Refer to his posting: http://coding.derkeiler.com/Archive/Ada/comp.lang.ada/2003-10/0587.html

PS: wxWindows changed name because on Microsoft demand (see http://wxwidgets.org/name.htm). May we still consider windows being a common word?

Konada.Db - Oracle Access Library

From: Frank Piron <frank@konad.net>
Date: Mon, 19 Apr 2004 13:09:02 +0200
Subject: Konada.Db
Newsgroups: comp.lang.ada

Konada.Db is an Ada95 library for easy and performant Access to Oracle Databases.

Konada.Db is built on top of Dmitriy Anisimkov's Ada Oci library, an Ada95 Binding to the Oracle Call Interface.

Some features of Konada.Db:

- Bind and Define Variables are managed by the library,
- Multiple Connections,

- Asynchronous Calls,
- Easy to use SQL interface to lob's - large objects
- Documentation and many Examples

Konada.Db is licensed under the GPL.

Download Konada.Db from: http://www.konad.de/download.htm

Reporting Compiler's Bugs

From: Sergey <cjkywt@hotmail.com>
Date: 15 Apr 2004 02:53:59 -0700
Subject: Re: Ada compiler
Newsgroups: comp.lang.ada

Ludovic Brenta wrote:

> > Ok, I don't know exactly but the only thing that I can said for certain - it isn't that good. I've tried to compile aws-1.4.0 using 3.2.3. Compilation finished well, but regression tests have badly failed...

> Have you filed problem reports in GCC's bugzilla? This kind of information, if detailed, is very valuable, especially if the problems are regressions since 3.15p.

I thought it doesn't make much sense, because everybody over there was (and still) saying that you shall stay away from using gcc-ada due to its pre-alpha quality.

> I just received a bunch of very detailed bug reports for 3.15p. Since 3.15p doesn't have a public bug database, I'll file them in the Debian bug tracking system for all to see. Time permitting, I'll do this over this and next weeks.

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 15 Apr 2004 12:41:48
Subject: Re: Ada compiler
Newsgroups: comp.lang.ada

[about reporting bugs in GCC 3.2.3 to bugzilla]

> I think that if you find a bug with 3.2.3, then you should check if the bug still exists in gcc CVS before reporting it to the gcc people. What is the point of telling the gcc developers about bugs they already fixed?

This is indeed the policy of GCC developers; they do not support any version of GCC but the latest release (3.3.3 as of now).

Before filing a bug to GCC, you should not only check it against the latest release, but also make sure that this bug was not introduced by the distribution; i.e. you should only file bugs if you use an unmodified GCC. Also, you should search the bug database to see if it has been reported before.

Still, if you find a bug in 3.2.3 that is fixed in 3.3.3, I think you should file it and close it immediately, and provide any workaround you know of for the version in question. The fact that this bug was fixed is valuable information that deserves to be public. The workaround, if any, is even more valuable.

> There is however a point in reporting such bugs to your distribution (Debian etc), since it is sometimes possible to backport fixes from CVS to gnat 3.15p or whatever

Yes. The distribution acts as the front line of support for all the software therein; if you find a bug in any software that came with your distro, you normally report it to the distro. The main reason for this is that distros often patch the software, and the bug may or may not originate from upstream.

If it turns out that the bug does come from upstream, the maintainer (for Debian that would be Matthias Klose or myself) will forward the bug to upstream, i.e. report@gnat.com or the GCC bugzilla database. If you *know* that the bug comes from upstream, then you help us by submitting the bug there as well as in the distro's bug database.

Another way in which bug databases are useful is to permit tracing patches to bugs; i.e. by looking at the bug log, you can find out which change to GCC fixed a particular bug. Without this information, it is very difficult to backport fixes.

I just filed a bunch of bug reports in the Debian database for GNAT 3.15p. There are several more that I have to file as well. The next steps are to try and reproduce them with GCC 3.2, 3.3 and 3.4. Those not fixed should then be tagged upstream and forwarded to GCC's bugzilla. I will appreciate any help with this; if you wish to contribute some of your time, please add information to the bugs in the Debian database: http://bugs.debian.org/cgi-bin/pkgreport.cgi?pkg=gnat

Quality of GCC 3.4

From: Georg Bauhaus <sb463ba@l1-hrz.uni-duisburg.de>
Date: Thu, 22 Apr 2004 11:57:29 +0000
Subject: Re: GCC 3.4 Ada compiler quality?
Newsgroups: comp.lang.ada

It has solved some problems for me having to do with private children, and with generic formals in certain circumstances.

I find -gnatwa quite useful

The project file support is better (though this might affect portabiliy).

From: Adrian Knoth <adi@thur.de>
Date: 21 Apr 2004 18:52:58 GMT
Subject: Re: GCC 3.4 Ada compiler quality?
Newsgroups: comp.lang.ada

I'd been using the 3.4-cvs for a while. Six months ago it was fine and did the job. I think is is worth upgrading, but I don't know why ;) IIRC they changed a lot in procedure-calls. Instead of pushing the parameters to the stack and popping them afterwards they're calculating the variable's address and do a normal subroutine-call afterwards. But consider this as highly iirc, I don't even know if it's true at all ;)

Someone contributed graph-coloring-optimization-code, I guess it's for automatically deciding which variables to use as register-vars (faster than RAM-vars). But I'm not sure ;)

Besides this a bugtracking-system exists and I'd guess the Ada-component itself was improved due to userfeedback. Dunno.

From: Duncan Sands <baldrick@free.fr>
Date: Thu, 22 Apr 2004 14:44:44 +0200
Subject: Re: GCC 3.4 Ada compiler quality?
Newsgroups: comp.lang.ada

Georg Bauhaus wrote: […]

> It has solved some problems for me having to do with private children, and with generic formals in certain circumstances.
> I find -gnatwa quite useful-
> The project file support is better (though this might affect portabiliy).

I have had problems with the code generated when using -gnatn, and sometimes when just using -O2 (mysterious segmentation faults, exception handling failing etc). I haven't noticed any regressions when compiling without optimisation, compared to gnat 3.15p.

The Development of GNAT

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 26 Mar 2004 00:07:29 +0100
Subject: Re: Is 3.15p -still- the latest GNAT 'p' release?
Newsgroups: comp.lang.ada

Dale Stanbrough wrote:

> I was looking at cs.nyu.edu the other day, and the latest Gnat version still seems to be 3.15, which dates from 2002.
> Is this -really- the latest release?

Yes, it is the latest official release, and is still recommended, unless you want to participate in the development of GNAT. [...]

The future is uncertain to me. ACT has said they plan to make additional "p" releases available, but there is no telling when. Meanwhile, a lot of work has gone into the upcoming GCC 3.4. It will be more stable, offer some features proposed for Ada 2005, as well as some additions to project files and the GNAT.* library. But it does not support tasking on powerpc-*-linux (a show stopper as far as I am concerned), and lacks GLADE.

My recommendation is: stick with 3.15p. If you want to be on the bleeding edge, experiment with GCC 3.4 or, better yet, get involved in the development of GCC 3.5. Avoid GCC 3.1, 3.2 and 3.3.

From: Robert I. Eachus
<rieachus@comcast.net>
Date: Sat, 27 Mar 2004 16:42:27 -0500
Subject: Re: Is 3.15p -still- the latest GNAT 'p' release?
Newsgroups: comp.lang.ada

> > > You could always contact friends who are (or work for) paying customers and get a Pro version from them.

> > This is pirating. I will never encourage such a disrespectful

> I thought Gnat was Open Source and was developed with taxpayer funds on that basis. No?

The situation is much more complex than that. First, having a non-public release of GNAT is not piracy, whether or not you paid for it. Paying ACT customers are encouraged not to redestribute the non-public releases in this way, but there is nothing that legally prevents them--or ACT from doing so. In fact, there have been times when I have been given access to such versions--by ACT--either to help decide whether some behavior was a bug, or to help fix a bug. In those cases, I have disposed of the wavefront (or in one case, I think, 3.12a release) once the problem was solved. I didn't do this for legal or moral reasons, but because I wanted any code I publically distributed to compile and run on the current GNAT public release.

On the other hand, if you as a GNAT licensee give a copy of the compiler to someone, then report their bugs and problems to ACT as your own, that is fraud.

Which brings us full-circle to the issue of public releases by ACT. First, AFAIK, any contracts involved in the creation of GNAT were between the government and New York University. NYU is, I think, a part owner of ACT. But the public releases of GNAT are intended as a public service, and if you check, you should download them from NYU, not from ACT. ACT is involved in deciding when a version of GNAT is stable enough to warrant becomming a public release.

If you are familiar with the concept of the Cathedral and the Bazaar, for some software the cathedral approach is much more appropriate. Public releases of compilers are one such case. If you want to put together several publically available software packages into a project, you are much better served if all of the developers of those packages used the same compiler. If each used different versions of GNAT, or required different versions of the same libraries, you end up having to do a lot more work.

So as far as I am concerned, it will be nice when there is a stable GNAT release that uses the new GCC backend. But I am quite content to use 3.15p until such a version is available. (Well, until the issue of Ada 2005 compatibility starts to become important. But right now, even the 2005 in that is just a guess.)

From: Stephen Leake
<stephen_leake@acm.org>
Subject: Re: Questions about Ada Core Technologies
Date: 06 Apr 2004 23:22:27
Newsgroups: comp.lang.ada

Ludovic Brenta wrote:

> I got some flak[1] about things I wrote in my draft Debian Policy for Ada [See also the relevant subsection in the Ada and Linux News section – su.]. While I prepare Draft 2, I would like to get some information in order to get some facts straight. Here is a list of questions I have; please answer them only if you have first-hand knowledge; I am trying to dispel rumours and spread facts :)
> - Will ACT make more "p" releases of GNAT in the future? They told me privately they would, but has anyone else heard about a public statement from ACT?
> [1] I mean I got ***friendly*** flak, kind of like error messages from an Ada compiler :)

The last I heard from them on this topic (about 6 months ago) was that they had not decided.

> Is it too much to ask for a release date or time frame?

Yes, it is too much to ask :)

> - Have ACT really switched their day-to-day development to the FSF?

I don't know. I also don't see how this is relevant to Debian.

> The changelogs suggest so, in which case I can suppose they merge selected changes to GNAT Pro in their private repository?

I assume they are maintaining at least a separate branch, if not a separate repository, but I really don't know.

> - Will the next GNAT Pro be based on FSF's GCC, or on ACT's private repository?

5.02a is current (released in March 2004). It is based on FSF gcc 3.2.3 (that's what gcc --version says).

As usual, ACT has made no firm statments about future plans.

> *- Does ACT recommend anyone switch to GCC instead of GNAT 3.15p? If so, which version of GCC?*

I believe they would say "test it with your application; use whichever is best for you".

> - Does ACT request that customers not distribute copies of GNAT Pro?

No. To be specific, there is nothing in the support contract that says this. They do point out that the non-public releases are non-public for a reason; they are more likely to contain bugs, and therefore should only be used with a support contract. Customers tend to agree with that position.

Again, I don't see how this is relevant to Debian.

> - Since GNAT Pro, as a derivative work from GCC, is necessarily distributed under the GPL, is the above request not an infringement of the GPL?

In short, the GPL says "If you give someone a binary, you also have to give them the source".

The GPL does _not_ say "if you give one person a binary, you also have to give everyone else the same binary". So even if ACT was requesting that customers do not distribute non-public releases, it would not be violating the GPL

The latest version of your Debian policy looks good to me, except for the part about ACT requesting non-distribution of non-public versions.

From: Steve <steved94@comcast.net>
Date: Wed, 07 Apr 2004 04:38:06
Subject: Re: Questions about Ada Core Technologies
Newsgroups: comp.lang.ada

Ludovic Brenta wrote:

> - Will ACT make more "p" releases of GNAT in the future? They told me privately they would, but has anyone else heard about a public statement from ACT? Is it too much to ask for a release date or time frame?

The last I heard is that they have no plans to stop making tese releases.

> - Have ACT really switched their day-to-day development to the FSF? The changelogs suggest so, in which case I can suppose they merge selected changes to GNAT Pro in their private repository?

Of course ACT will be maintaining their own site. They have customers that depend on them and they cannot tolerate instability.

A few months ago there was an announcement on the gcc mailing list that ACT would be updating the FSF repositoriy on a regular basis. If I recall correctly, Arnauld Charlet of ACT was made responsible for keeping the FSF repository up to date.

> - Will the next GNAT Pro be based on FSF's GCC, or on ACT's private repository?

ACT's private repository. Over time the difference between the two will continue to decrease.

> - Does ACT recommend anyone switch to GCC instead of GNAT 3.15p? If so, which version of GCC?

ACT doesn't recommend anyone use a 'p' release for serious work. The public releases are primarily released for academic use. Of course the 'p' releases are of good quality and could be used for serious work, but ACT does not stand behind them. For that you need to by GNAT Pro.

From: Florian Weimer <fw@deneb.enyo.de>
Date: 07 Apr 2004 17:27:47 +0200
Subject: Re: Questions about Ada Core Technologies
Newsgroups: comp.lang.ada

The main development happens on ACT's infrastructure (after all, they have a far more extensive test suite (which is partly covered by NDAs and cannot be published), an automated regression tester based on that test suite, a separate bug tracking system, their own developer communication channels &c).

Arnaud Charlet regularly (daily?) merges ACT's changes into the FSF tree. But this doesn't mean that this tree is the sole reference for development.

From: Robert I. Eachus <rieachus@comcast.net>
Date: Tue, 13 Apr 2004 16:54:03 -0400
Subject: Re: Questions about Ada Core Technologies
Newsgroups: comp.lang.ada

Stephen Leake wrote:

> As I said, there is nothing in the customer contract with ACT about not distributing GNAT. But they do informally request it.

Actually, what ACT requests/recommends is not distributing wavefront versions. Wavefront versions are usually a response to a particular problem encountered by a (supported) user. As such they are less thoroughly tested than other versions of GNAT.

What about the versions ending in a, not w? They are betwixt and between. In general if an a version gets to where it has a short buglist and no outstanding major problems, it gets converted to a p (public) version pretty quickly. What has been going on recently, as I understand it, is that ACT has versions which perform well on some platforms but fail on others. When you have customers with a support contract, you can insure that they have a variant that runs on the hardware (and OS) that customer uses.

Once the 5.xx versions become stable, ACT will stop recommending that the public use 3.15p. Notice that the 5.xx versions are completely available to the public, just not recommended unless ACT has insured that they work in your environment. (And of course they are under no obligation to test those versions on non-supported hardware and OS configurations.)

As for me, I have been staying with 3.15 because I want the software I write to be as widely distributable as possible. (I just went through a painful process with a failing disk. If it had failed completely I would have just gone to my last complete backup. But when it started failing I was able to do one last incremental backup--but it took days. Then I had to build a new disk, and apply all the increments. Since that disk held the partition I use for Ada development, I didn't want to change any libraries in the middle of all this.)

GNADE 1.5.2 - GNAT Ada 95 Database Development Environment

From: Samuel Tardieu <sam@rfc1149.net>
Date: Wed, 17 Mar 2004 09:16:40 +0100
Subject: Re: Release of GNADE 1.5.2
Newsgroups: comp.lang.ada

[See also same topic in AUJ 24.3 (Sep 2003), and AUJ 24.1 (Mar 2003), p.14 -- su]

The objective of the GNADE project is to provide various packages which are allowing Ada 95 applications to access relational data base products, mainly SQL RDBMS.

This release provides the first time a common build procedure for windows and *unix systems. Additionaly the use of the GPS system of ACT.com has been taken into account, which means for each component a gps project files has been created.

http://gnade.sourceforge.net/

Mneson - Database System

From: Marius Amado Alves <maa@liacc.up.pt>
Date: Tue, 23 Mar 2004 20:39:00 +0000
Subject: Mneson: persistent untyped graphs
Newsgroups: comp.lang.ada

I'm glad to announce the first release of Mneson, a 100% Ada library for persistent untyped directed graphs of basic values (integer, string, float).

http://www.liacc.up.pt/~maa/mneson

It's open source.

From: Marius Amado Alves <maa@liacc.up.pt>
Date: Wed, 24 Mar 2004 13:01:01 +0000
Subject: Re: Mneson: persistent untyped graphs
Newsgroups: comp.lang.ada

[Regarding SDC Conditions, the licensing terms of Mneson, http://www.liacc.up.pt/~maa/mneson]

If it's non-commercial, just use it.

> I don't find any statements about this license.

There is some amount of discussion and commentary in SDC and OSI fora.

> Neither is it on the OSI list as I can see.

It is open source, but no, not OSI-compliant. The SDC license breaches clause 6 of the OSD under a certain conservative interpretation of that clause, namely one whereby "restrict" means "requiring a separate deal". It seems this is the official interpretation by OSI. I'm liberal so I call it open source. It does not hurt any other clause of the OSD.

The commercial-open source clash is still an open issue. Never mind. Just use it. If it's commercial, tell the authors. They'll cut you a fair deal. You'll not have to pay unless you get rich, and then it doesn't hurt you. That's the gist of it. Ignore the legalese. "Just do it."

Lex and Yacc for Ada

From: Wurbel Eric <wurbel@univ-tln.fr>
Date: Wed, 24 Mar 2004 15:40:49 +0100
Organization: Universite de Toulon et du Var
Subject: unable to find a downloadable of aflex/ayacc
Newsgroups: comp.lang.ada

[See also same topic in AUJ 23.4 (Dec 2002) -- su]

I'm in search of a lex/yacc equivalent for Ada. After an extensive search on the net, The only tool I found is aflex/ayacc, but I am unable to download it (the server ftp://liege.ics.uci.edu/ seems to reject anonymous connections).

Any clues?

From: Jeff <jeff.huter@bigfoot.com>
Date: 24 Mar 2004 14:04:10 -0800
Subject: Re: unable to find a downloadable of aflex/ayacc
Newsgroups: comp.lang.ada

> I dont read French, but ayacc/flex is mentioned here and appeare to be accessable. http://perso.wanadoo.fr/pascal.obry/contrib.html

There is also a version at: http://www.macada.org/tools.html with PDF documentation.

From: Craig Carey <research@ijs.co.nz>
Date: Tue, 30 Mar 2004 01:58:22 +1200
Subject: Re: unable to find a downloadable of aflex/ayacc
Newsgroups: comp.lang.ada

The directory has gzipped "Mach-O executable ppc" documents instead of Ada 95 files (so implies my GNU Cygwin 'file.exe' program). It might not be the latest and so lack the 'UMASS' extensions.

There seems to be a merged edited version of the Yacc Ada parser in this file (also Adgoop was updated): http://www.ijs.co.nz/code/ada95_adagoop_parser.zip

AYacc would have user's Ada code be inside of a case statement inside of a loop statement.

Adagoop would instead create Ada code that creates a tree on the heap having nodes that are tagged records, (one for each main Yacc entry).

Adasubst & Adadep

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Fri, 27 Feb 2004 13:24:56 +0100
Organization: Adalog
Subject: New version of Adasubst/Adadep
Newsgroups: comp.lang.ada

Adasubst 1.2

Added overloaded form of enumeration literals in the dictionary Fixed bug not allowing parameterless functions overloaded form. Fixed bug with -P option to process only the visible parts of nested packages/tasks/protected

Adadep 1.2

Added number of uses and declaration kind of each entity in the report.

Download, more information, from Adalog's components page: http://www.adalog.fr/compo2.htm

ASIS for GNAT: New Project and First Versions

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Fri, 12 Mar 2004 17:05:07 +0100
Subject: New Project: ASIS for GNAT
Organization: AdaCL
Newsgroups: comp.lang.ada

I like to announce the beginning of a new Project.

The Project aims to provide an up to date implementations of ASIS = Ada Semantic Interface Specification for GNAT based on gcc 3.5. The currently available open source ASIS implementation is for GNAT on gcc 2.8.1.

The Project is located at sourceforge: http://sourceforge.net/projects/gnat-asis/

Current experinental versions are available via cvs.

From: Arnaud Charlet <charlet@gnat.com>
Date: Tue, 16 Mar 2004 14:59:28 +0100
Subject: Re: New Project: ASIS for GNAT
Newsgroups: comp.lang.ada

Note that having a reliable ASIS working with a particular GNAT version is a lot of work, and certainly much more than just putting some sources together as you described, since there's a very close relationship between the GNAT and ASIS data structures. I'm afraid your resulting source package will suffer from the same trouble gcc 3.3 did: frozen Ada sources with a moving (incompatible) GCC back-end that compiles but is pretty unreliable.

Same comment for glade btw: the glade run time is tightly coupled with the corresponding GNAT run time.

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Sun, 14 Mar 2004 13:04:43 +0100
Organization: AdaCL
Subject: ASIS for GNAT first release
Newsgroups: comp.lang.ada

I am pleased to say that the ASIS for GNAT effort was successful and I have released a first snapshot version for gcc 3.5.

The Project is looking for maintainers of other versions (gcc 3.4 and gcc 3.3.1). It's not that difficult to add A.S.I.S. once you have compiled your own gcc.

From: Martin Krischik <krischik@users.sourceforge.net>
Subject: ASIS for GNAT 20040403 release
Date: Sat, 03 Apr 2004 13:10:07 +0200
Organization: AdaCL
Newsgroups: comp.lang.ada

New snapshot release from A.S.I.S. for GNAT.

This Version contains gnat sources for GCC 3.5, GCC 3.4 and GCC 3.3.1. The first two from today.

The following A.S.I.S. tools are bundled: adabrowse, adadep, adasubst, asistant, gnatelim, gnatstub.

Binary package for i686, GCC 3.5, SuSE 9.0 available.

Delphi to Ada Translator

From: Jacob Sparre Andersen <sparre@nbi.dk>
Date: 19 Feb 2004 21:44:44 +0100
Subject: Re: Delphi to Ada translator
Newsgroups: comp.lang.ada

Szymon Guz wrote:

> Do you know any Delphi to Ada translator?

There is at least a Pascal to Ada translator, which also (or in a special edition) accepts Borland Pascal source code. It may also have been modified to accept Delphi source code.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 19 Feb 2004 14:53:00 -0600
Subject: Re: Delphi to Ada translator
Newsgroups: comp.lang.ada

PasTran does Pascal-to-Ada, of course, and we've done custom versions for various clients and various Pascals, but we've never had anyone ask for Delphi. We could of course do a custom version, but they would need to be a quite a bit of code to translate to make it economic.

From: Gautier
Date: Thu, 19 Feb 2004 22:26:27 +0100
Subject: Re: delphi to ada translator
Newsgroups: comp.lang.ada

P2Ada for instance (with the BP2P preprocessor). NB: OO support is ongoing.

www.mysunrise.ch/users/gdm/gsoft.htm

From: Leif Holmgren
Date: Mon, 23 Feb 2004 22:35:51 +0100
Subject: Re: delphi to ada translator
Newsgroups: comp.lang.ada

Craig Carey wrote:

> Has anybody succeed in putting Delphi code inside of a DLL and getting it called by a GNAT Ada 95 main program (inside of a *.exe file) ?.

Yes, I have about 2 years ago, using Delphi 3 and gnat 3.15a1. It took a while to figure out how to generate the files required by gnat. (.lib-files?). All the tools needed was available with gnat. I also think I remember […]

> Any URLs saying how to put Delphi GUIs inside of DLLs?

No. Did this with Delphi 1 in 1995 (called from a VB3 app). Never caused any problems. Autocreate of forms naturally won't work. You create an procedural interface that call the form-stuff inside the DLL.

You should also be able to create an ActiveX interface that you can use from Ada, but all my attempts to do this has resulted in loads of unreadable code that worked but was not reliable. I guess it leaked memory but I could never find the leaks. Do a web-search for a package called gnatcom. It contains the stuff you need for Ada/COM interfacing.

Sorry for not beeing able to be very precise. I really don't remember how I did most of this stuff.

NaturalDocs - Source Code Documentation Generator

From: Samuel Tardieu <sam@rfc1149.net>
Date: Fri, 16 Apr 2004 13:25:18 +0200
Subject: NaturalDocs
To: ada-france@ada-france.org

NaturalDocs est un outil libre permettant de générer automatiquement de la documentation pour un grand nombre de langages (similaire à Doxygen pour ou Javadoc), dont Ada.

http://naturaldocs.org/about.html

Playing Ogg Files

From: Preben Randhol <randhol@pvv.org>
Date: Fri, 5 Mar 2004 09:30:27 +0000
Subject: Re: Ogg and Ada
Newsgroups: comp.lang.ada

Jorge Suárez-Solis Rivaya wrote:

> I've read in an old post (year 2002) about playing ogg files from Ada programs. Does anyone know how can I do it?

As far as I know there isn't an Ada binding yet.

But Chad suggested earlier to: Take a look at the AdaSDL binding (thick or thin). If I remember correctly SDL should be able to play Ogg files, but I have never tried it.

http://www.libsdl.org

Ada-MIDI Library

From: Patrice Freydiere <frett27@free.fr>
Date: Mon, 12 Apr 2004 16:36:50 +0200
Subject: Ada-MIDI library new release
Newsgroups: comp.lang.ada

A new release of the native MIDI format library for Ada has been posted on http://pfreydiere.free.fr

Feel free for feedbacks

Optical Ballot Scanner

From: Tom Moran <tmoran@acm.org>
Date: Tue, 09 Mar 2004 02:32:25 GMT
Subject: Ada optical ballot scanner app
Newsgroups: comp.lang.ada

I've posted an optical ballot scanner program at http://home.comcast.net/~twmoran

It was developed to process images of a thousand cards marked by kids last Christmas to indicate their toy etc preferences, for a Christmas toys-for-needy-kids undertaking (www.mytwofrontteeth.org) The images were scanned by an inexpensive batch scanner. It uses an FFT-based algorithm based on sample unmarked and fully marked ballots so it can do alignment and finding of checkboxes without need for more customization. Source for the main program is also posted, with no restrictions.

AdaCL 4.1.0 - Ada Class Library

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Sat, 03 Apr 2004 13:05:20 +0200
Organization: AdaCL
Subject: AdaCL 4.1.0 released.
Newsgroups: comp.lang.ada

Improvements on existing features.

AdaCL.CGI now support "multipart/form-data" - that allows file upload.

Support for dynamic link libraries - this will allow the creation of a Debian package.

gnatprep has been removed. Conditional compile now done by use of seperate directories - this makes it easier to use the lib if you still want to link staticly.

Abstract:

AdaCL provides library services for scriptig in Ada:

- A text search and replace library.
- A complete cgi binding.
- Execution of external programms inclusive I/O redirection with

Ada.Text_IO. Unlike GNAT.OS_Lib AdaCL lets you wait on a given asynchronous process instead of just the first to end.

- A garbage collector.
- Booch components for indefinite elements.
- Trace feature - very handy for CGI (no debugger, no console output).
- Control cdrecord and makeisofs.
- Some cvs tools.

http://www.ada.krischik.com

GLADE For GNAT: New Project

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Sat, 03 Apr 2004 13:13:14 +0200
Organization: AdaCL
Subject: New Project: GLADE for GNAT
Newsgroups: comp.lang.ada

I like to announce the beginning of a new Project.

The project whishes to providing an up to date implementations of Annex E (Distributed Systems) for GNAT.

The Project is located at sourceforge: http://sourceforge.net/projects/gnat-glade/

Current experinental versions are available via cvs.

Since GLADE is a lot more difficult then ASIS help would be apreciated.

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Fri, 16 Apr 2004 08:36:02 +0200
Organization: AdaCL
Subject: Re: New Project: GLADE for GNAT
Newsgroups: comp.lang.ada

Dr. Adrian Wrigley wrote:

> How does this relate to the ACT GLADE releases?
> If ACT are working with their own sources, fixing bugs for paying customers, providing them with wavefront releases, won't the "Sourceforge GLADE" always be out of date? Every time a new release is made by the original authors, you will have to integrate their changes. And will it be possible to take the project in a new direction without the sources diverging, giving a permanent split?

Well, the paying customers are allways up front. What nags me is that ACT has nor released a "p" for 2 years. So the 3.15p compiler becomes out of date. Only today gcc 3.4 on the gnat list (gnatlist@lyris.seas.gwu.edu) was suggested for use with G5 PPC.

> I am a constant user of GLADE, but have encountered a major bug. Under certain (common) circumstances, failure of one partition can cause the whole program to lock-up - even when the appropriate options are specified in the build. I have not been able to get hold of a wavefront release where this is (said to be) fixed. (all this on versions 3.15p, Intel Linux). Does anyone know how I can get hold of a more recent GLADE for 3.15p?

There are no more recent releases for 3.15p. All more recent releases are for paying customers and GNAT 5.x.

The bad news is: There might never be since act plans a new Annex E based on PolyORB.

While the sources are GPL there are only distributed to paying customers. The GPL says that you must provide the sources to anyone you provide binaries to.

Without a public binary release act does not need to provide a public source release. :-(.

> Good luck with the project though! I'd love to see more people using the technology, which beats the alternatives by a long way (in terms of ease of use and flexibility).

The current cvs version compiles and links. However there seems to be a chance in compiler options so some .o files are not generated. So more work is needed.

Strings_Edit 1.4

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Sat, 07 Feb 2004 11:20:34 +0100
Subject: Strings_Edit v1.4
Newsgroups: comp.lang.ada

The package Strings_Edit provides I/O facilities. The following I/O items are supported by the package:

* Integer numbers (generic, package Integer_Edit)
* Floating-point numbers (generic, package Float_Edit)
* Roman numbers (the type Roman)
* Strings

http://www.dmitry-kazakov.de/ada/strings_edit.htm

The new version fixes bugs in processing very large numbers.

Crytographic Library

From: Jano <nono@unizar.es>
Date: Sun, 1 Feb 2004 17:35:00 +0100
Subject: Re: Free cryptographic library for Ada?
Newsgroups: comp.lang.ada

Peter C. Chapin wrote:

> I am designing a security sensitive application and I would like to use Ada to develop it. This is something of a hobby project so it is very low budget. In my application I need to make secure hashes and digital signatures along with the necessary key management that implies. I'm looking for a free (GPL or similar license) crypto library written entirely in Ada. I don't particularly want to link to a C library since part of the point of using Ada in this case is

specifically to avoid C.
Does such a library exist? I've been prowling around on the 'net but so far the projects I've turned up are very incomplete and inactive looking.

It's quite old but I've been using its SHA1 and Tiger implementation without any problem. It has other digital signatures as well: http://sourceforge.net/projects/adacf/

From: Jeffrey Carter <jrcarter@acm.org>
Date: Sat, 31 Jan 2004 20:10:58 GMT
Subject: Re: Free cryptographic library for Ada?
Newsgroups: comp.lang.ada

There's an Ada implementation of the Solitaire algorithm at: www.schneier.com/code/solitaire-ada.zip

Package Debug 2.2

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Tue, 10 Feb 2004 13:18:38 +0100
Organization: Adalog
Subject: Package Debug V2.2 available
Newsgroups: comp.lang.ada

I just released a new version of package Debug, which contains very useful tracing functionalities.

Improvements include:

- Binary tracing of any type (using streams).
- Tracing objects (traces are performed through Initialization/Finalization), great for tracing functions that return complex expressions
- New package Debug.Assert. Adds a conditionnal trace (tracing only if some assertion fails) and a reentrancy detector object (traces when two tasks call the same subprogram at the same time). Note that this is something nearly impossible to check with a debugger.
- changed license to GMGPL (it was already available freely for any purpose, but it seems easier to use a well known license than trying to invent some new wordings).

Bug fixes:

None.

This package has been publicly available since 1999. We never received a bug report.

Availability:

See Adalog's components page: http://www.adalog.fr/compo2.htm

Ada Syntax Scripts for Vim

From: Preben Randhol <randhol@pvv.org>
Date: Fri, 30 Apr 2004 18:58:20 +0000
Subject: Re: Recommendations??
Newsgroups: comp.lang.ada

Larry Hazel wrote:

> I have always used vi for writing code and would still be comfortable with that.

I strongly recommend that you use vim (VI iMproved). I have some unfinished Ada scripts that can be helpful if you want a copy. http://www.vim.org/

I use Vim and here is a screenshot how it looks like when I code with my setup: http://www.pvv.org/~randhol/Vim/project-ada.png

(Picture is from Linux, but you can get the same behaviour in Windows)

I use these excellent scripts, all under: http://vim.sourceforge.net/scripts/ :
script.php?script_id=31
script.php?script_id=69
script.php?script_id=58
script.php?script_id=21

If you prefer a different IDE there is also: script.php?script_id=95
but I like the project approch more.

Controlling the Serial Ports

From: Michael Bode <m.g.bode@web.de>
Date: 10 Mar 2004 22:28:23 +0100
Subject: Re: Serial port control Windows and Linux
Newsgroups: comp.lang.ada

Erlo Haugen wrote:

> I know that questions like this have been asked before. I did some searching on Google and found som links, but they were all broken.
> I need to control a serial port from an Ada program. I need to have control of the handshake signal (set and clear). Any good pointers to packages/bindings??

Maybe http://home.t-online.de/home/michael_bode/eigener_mist_en.html is useful for you.

From: Jerry Petrey <jdpetrey@raytheon.com>
Date: Tue, 09 Mar 2004 16:01:45 -0700
Organization: Raytheon Company
Subject: Re: Serial port control Windows and Linux
Newsgroups: comp.lang.ada

I have used Stephe Leake's com port utilities with great success (with some modifications) to control the serial ports on a Windows platform. They can be found here: http://www.toadmail.com/~ada_wizard/ada/win32_com_ports.zip

Ada Unit Test Tool

From: Simon Wright <simon@pushface.org>
Date: 14 Feb 2004 17:11:32 +0000
Subject: Re: debbuger or unit test tool
Newsgroups: comp.lang.ada

Orvio wrote:

> In other case, can anyone suggest me another tool for Ada unit test?

We are happy with AUnit <http://libre.act-europe.fr/aunit/>. We did evaluate VectorCast, I don't remember the details but my _impression_ was that the possible advantage of being able to change the test script without recompilation was not so big an advantage on fast hardware. Also we had trouble getting it to understand our (complex GNAT) compilation environment.

Ada-related Products

ACT - GNAT Pro 5.02a

From: Cyrille Comar <comar@act-europe.fr>
Date: Thu, 26 Feb 2004 18:25
Subject: Announcing Availability of GNAT Pro 5.02a
To: eu-announce@act-europe.fr

The 5.02a release completes the transition from GNAT3 to GNAT5 for most of the GNAT Pro configurations. This is a comprehensive enhancement to the GNAT Pro technology offering multi-language capabilities: Ada, C, C++ (contact our sales department for C and C++ support options). It also offers many new features […] both in the core GNAT compiler technology and in the programming environment toolset. For example:

- greater flexibility in the configurability of the High Integrity Runtimes,
- fewer restrictions on component representation clauses,
- a new tool 'gnatclean' for removing compiler generated files,
- significant enhancements to gnatpp and gnatelim,
- new advanced versions of the gtkada, glade and asis add-ons

5.02a contains the latest release of the GNAT Programming System, GPS 1.4.1, which adds capabilities such as:

- enhanced editor and source navigation
- improved debugger support
- better integration under Windows
- more powerful customization capabilities
- support for duplicated Ada file names within a project

We encourage you to install and start using this latest version of the GNAT Pro tool suite. As always, for questions, or to inform us of issues that you encounter, please let us know through the GNAT Tracker report facility or by email at the usual report@gnat.com address.

Looking ahead to future releases, the next major release will be 5.03, which will be available some time early in 2005. This again will contain major new functionalities, including much more extensive implementation of new features for the new version of the Ada standard.

In addition, later this year, we will, if needed, produce a follow-up release to 5.02a, called 5.02a1 that will correct any significant deficiencies found in the 5.02a release. This will be a maintenance release only and will not contain any new functionality.

AdaCore announces local distributor in Australia

Date: Mon, 19 Apr 2004
URL: http://www.gnat.com/pressroom_09.php

New York, NY - Monday, April 19, 2004 - AdaCore, leader in Ada 95 technology and providers of the GNAT Pro Ada development toolsuite, today announced that it has entered into a distribution agreement with Dedicated Systems Australia Pty Ltd, a distributor providing Australia and New Zealand with world-class software and hardware solutions in the real-time and embedded computing markets with a special focus on defense and communications. This collaboration will not only provide our Pacific Rim customers with a fully integrated Ada solution for native and cross systems, but also bring them the high level of customer attention that is the hallmark of AdaCore.

"We are pleased to be working with Dedicated Systems," said Robert Dewar, President of AdaCore. "This collaboration allows us to continue the tradition of partnering with our customers and maintaining a close relationship long after the point of sale."

"AdaCore has a fantastic reputation as a technology leader, providing products and outstanding customer support. We are excited to work with them in our territory" commented John Salerno, Director of Software Engineering at Dedicated Systems.

About Dedicated Systems

Dedicated Systems Australia is a distributor in the Australia and New Zealand Real-Time and Embedded Computing Markets with a special Focus on Defence, Communications and Automotive applications. They represent world class hardware and software vendors to bring high quality and synergetic products to customers. Their headquarters are in Adelaide, South Australia. Key software suppliers are: Windriver Systems, Ada Core Technologies, Programming Research, RTI and KUKA Controls. Key hardware suppliers are: ELMA, Schroff, Condor Engineering, VMETRO, Sky Computers and Themis.

About AdaCore

AdaCore, a privately held company founded in 1994, with major offices in New York City and Paris, produces and supports the GNAT Pro family of open-source Ada 95 software development environments. Based on the GNU GCC technology, GNAT is available on more platforms than any other Ada compiler and is the only implementation of the complete Ada 95 language. GNAT Pro, the professional edition of the GNAT technology and the premier Ada development environment on the market, is used on enterprise-critical projects encompassing areas such as low-level communications control, high-integrity real-time applications and large-scale distributed systems.

ACT - Ada95 Development Environment for SGI ALTIX Product Line

Date: Tue, 20 Apr 2004

URL: http://www.gnat.com/pressroom_10.php

New York, NY.

AdaCore, the leader in Ada 95 solutions and providers of the GNAT Pro Ada development tool suite, today announced it is expanding its support agreement with Silicon Graphics. The new agreement will include the development and support of the GNAT Pro product line on the SGI® Altix® family of superclusters and servers, which feature the 64-bit Intel® Itanium® 2 processor and the Linux® operating system.

Since its launch just over a year ago, SGI Altix has become the first Linux OS-based system to scale to 256 Itanium 2 processors in a single system image-and thousands more via clustering-using the powerful SGI® NUMAflex global shared-memory architecture. The Altix architecture handles large data sets with ease, giving software developers an opportunity to provide 64-bit Linux applications to customers in manufacturing, oil and gas exploration, homeland security, earth and environmental sciences research, and life sciences.

SGI has a long history of building high-performance computing systems that perform well with Ada applications, and has worked with AdaCore for nearly a decade. The SGI Altix server port is the latest and most exciting chapter of this story.

AdaCore will continue to provide its Premium level service to all SGI customers who purchased Ada 95 compiler support contracts for the MIPS® processor and IRIX® OS-based product family and will be developing technology to aid those customers choosing to transition from this processor architecture to the new Itanium 2 processor-based Altix family.

Wind River Teams with AdaCore on Platform Safety Critical ARINC 653 for Use in Boeing 7E7 Common Core System

Date: Tue, 20 Apr 2004
URL: http://www.gnat.com/pressroom_11.php

New York, NY and Alameda, Calif.

Wind River Systems, Inc., the global leader in device software optimization (DSO), and AdaCore, the global leader in Ada 95 solutions and providers of the GNAT Pro Ada development tool suite, today announced that AdaCore's GNAT Pro High-Integrity Edition for AE653 has been selected as the Ada environment for the Wind River Platform for Safety Critical ARINC 653, to be used in the Boeing 7E7 Dreamliner Program.

The Wind River Platform with AdaCore's technology will be used on the 7E7's Common Core System, provided by Smiths Aerospace, which is the backbone of the airplane's computers, networks and interfacing electronics. The 7E7 Common Core System comprises approximately 80 to 100 applications running simultaneously which will control many of the airplane's avionics and utilities functions.

Wind River Platform for Safety Critical ARINC 653 is part of Wind River's Safe and Secure Program, which provides fully integrated device software platforms to meet the FAA's stringent safety certifications. AdaCore's GNAT Pro High Integrity for AE653 was developed specifically for Wind River Platform for Safety Critical ARINC 653. Based on the Ada programming language, Ada Core's development tools have been used extensively in recent years for safety-critical avionics applications because of its high levels of portability. With AdaCore's GNAT Pro High Integrity now built into Wind River's Platform for Safety Critical ARINC 653, Boeing and its other 7E7 suppliers will be able to leverage their existing software investments and easily adapt them to be used in the new Common Core System architecture.

Designed to be the most advanced and efficient commercial airplane in its class and predicted to dramatically alter the future of avionics, the Boeing 7E7 Dreamliner will set new standards for environmental responsibility and passenger comfort. Developed to seat 200- to 250-passengers, the 7E7 will fly between 3,500 and 8,500 nautical miles at the same speeds as today's fastest twin-aisle commercial airplanes - the 777 and 747. Boeing began offering the new 7E7 to customers in early 2004 and is projecting that it will sell up to 3,500 7E7s over the next 20 years.

ACT - GPS 2.0.0

AdaCore is pleased to announce the immediate release of GPS 2.0.0.

GPS, the GNAT Programming System, is an advanced IDE that streamlines your software development process - from the initial coding stage through testing, debugging/verification and maintenance. Designed by programmers for programmers, GPS is a new kind of IDE that offers the experience of designing software in a uniquely comfortable environment.

The 2.0.0 version is a major release and features many improvements, notably:

- A more powerful source editor including the addition of many source navigation capabilities
- Speed improvements
- Improved customization, in particular:
  - A generic version control interface that can be tailored to your specific CM tools and established procedures
  - A Python interpreter integrated in GPS under GNU/Linux, Solaris and Windows for powerful scripting and extensions
- Improved font rendering under Linux and Solaris
- Support for the IRIX platform
- C/C++ improvements
  - Better handling of C++ classes in entity browser
  - Better handling of C/C++ builds.

AdaCore and Ada 2005

As part of the ongoing standardization activities for Ada, the language is reviewed periodically to see if corrections or new features are warranted. This process is carried out under the auspices of the International Organization for Standardization ("ISO"), more specifically by the Ada Rapporteur Group ("ARG"), a technical committee from ISO's WG9 (the Working Group for Ada). The ARG includes Ada compiler implementors, users, and other language experts. The ARG is currently working on a set of proposed amendments to Ada 95 that are likely to become part of a revised language standard scheduled to be completed sometime in 2005.

AdaCore is directly involved with the Ada 2005 language amendment process, not only by participating in the ARG, but also by implementing / testing the new feature proposals as they become stabilized. We are also developing and submitting candidate conformance tests as the new features are implemented.

The Ada 2005 development process has several main goals. One aim is to remove limitations and to increase the usefulness of existing features. Other objectives are to provide enhanced capabilities for object-oriented programming and to better address the needs of the real-time and high-integrity communities. There is also the desire to standardize on certain common implementation extensions (e.g., pragma Assert) and to expand the set of predefined library units. Of course, upward compatibility is an important requirement.

Following is a short overview of a few of the key features proposed for the next revision of Ada.

Mutually Dependent Package Specifications

Ada users have long been asking for a way to have mutually dependent types that can be declared in independent packages. A new form of "with" clause, called a "limited with" clause, is proposed to address this need. The "limited with" clause allows packages to create weak dependences on one another, essentially providing an incomplete view of each package's types. This permits two or more separately compiled types to reference each other via access components. "Limited with" clauses have a natural implementation in source-based compilers such as GNAT, and support for this feature is already available in current versions of GNAT.

Aggregates for Limited Types

As an example of relaxing restrictions to make existing Ada 95 features more useful, there is a revision proposal to allow aggregates to be given for limited types. Limited aggregates would be permitted in contexts where a copy into a preexisting object is not required, such as an actual parameter or in the initialization expression of an object declaration. This feature has been implemented in GNAT.

More Contexts for Anonymous Access Types

Another proposed feature enhancement is to allow the anonymous access type notation (used in Ada 95 for access parameters) to appear in more contexts, permitting constants and record components to be designated as "access". This will help to reduce the need for explicit access type conversions. This feature has been implemented in GNAT.

Java-like Interfaces

For the object-oriented domain there is a proposal for a simplified multiple inheritance mechanism. During the design of Ada 95's OOP features there was a conscious decision to avoid multiple inheritance, because of concern with the distributed cost that would be imposed on the use of single inheritance. Since that time, a restricted form of multiple inheritance, sometimes called inheritance of interfaces, has been used to good effect in languages such as Java. This kind of inheritance does not impose the overhead that is incurred by full-blown multiple inheritance in languages such as C++. For supporting this in Ada 2005, the proposal is to have a new form of type called an interface type, similar to an abstract tagged type with no components. A tagged type will be able to inherit from multiple interface types, overriding the operations inherited from each of its parent interfaces.

Real-Time and High-Integrity Support

The ARG is also evaluating several proposals in the real-time and high-integrity areas. These include support for task termination handlers, timing events, execution-time clocks, alternative task dispatching policies, and standardization of the Ravenscar restricted tasking profile.

New Pragmas

Several pragmas that have been supported by GNAT for some time are candidates to be added to the Ada standard. Specifically, pragmas Assert, Unsuppress, No_Return, and Unchecked_Union are all viewed as being generally useful and important to standardize to improve portability of Ada programs.

Predefined Library

Extensions are also being considered for the predefined Ada library. Some of the new packages being proposed are in the areas of directory operations, matrix and vector operations, sockets, and for supporting some of the candidate features for the real-time domain. There is also interest in defining a standard set of container libraries, although that may be addressed by a separate standards effort, with some guidance from the ARG.

AdaCore is committed to the next version of Ada, and plans to continue implementing the new features as the proposals are stablized and approved for the Ada 2005 revision, as well as to prioritize our efforts according to expressed user interest. Prior to official standardization, these additions to the GNAT Pro technology can be activated using a special switch (-gnatX) provided specifically for enabling experimentation with the new features. Please stay tuned for the implementation of additional Ada 2005 features to GNAT in the coming months.

IPL Announces AdaTest Integration with GPS

From: "Jamie Ayre" <ayre@act-europe.fr>
Date: Tue, 1 Jun 2004 09:59:49 +0200
To: Dirk Craeynest <Dirk.Craeynest@cs.kuleuven.ac.be>
Subject: RE: ACT's press releases

Bath, UK, April 5 2004 - IPL Information Processing Ltd, suppliers of software testing tools announced that its AdaTEST 95 product has been integrated with the GNAT Programming System (GPS) from AdaCore.

AdaTEST 95 is a tool which allows users to test Ada software thoroughly and efficiently. In addition to the facility to generate test drivers and programmable stubs, AdaTEST 95 users can also measure test coverage and check static analysis metrics. Using Test Support Packages, users can automatically check global data for unexpected corruption.

GPS, the GNAT Programming System, is an advanced IDE that streamlines the software development process - from the initial coding stage through testing, debugging/verification and maintenance. Designed by programmers for programmers, GPS is a new kind of IDE that offers the experience of designing software in a uniquely comfortable environment.

The integration of AdaTEST 95 into GPS means that users can invoke a range of AdaTEST 95 facilities from within the IDE.

"We are confident that GNAT Pro/GPS users will appreciate the fast access to AdaTEST 95 which this integration gives them," says Ian Gilchrist of IPL. "It is always a challenge to improve productivity in the area of software testing. We believe this new facility will significantly lighten programmers' workloads."

"We are delighted that IPL has been able to rapidly integrate AdaTEST 95 into GNAT Pro / GPS," says Arnaud Charlet, AdaCore's GPS Product Manager. "This demonstrates the ease of extending GPS and tailoring it to add new tools, so users can adapt our IDE to their own needs."

Further information: AdaCore, www.act-europe.com

ARTiSAN - ARTiSAN's Real-time Studio Selected by BAE SYSTEMS

BAE SYSTEMS selected Real-time Studio for object oriented UML modeling of complex systems on Hawk program due to tool's flexibility, enabling customization for full SPARK Ada support.

Cheltenham, UK February 18, 2004 ARTiSAN Software Tools, a global leader for UML-based, real-time systems and software modeling tools, today announced that BAE SYSTEMS, a recognized leader in the field of systems, defense, and aerospace engineering, has selected ARTiSAN's Real-time Studio Professional as their tool of choice for the object oriented modeling of complex systems in UML on the Hawk program primarily due to a key differentiator of the tool - its extensibility.

"BAE SYSTEMS has been using Real-time Studio successfully at a number of sites for some years. Software engineers on the Hawk Project use RtS to express their implementation of a module support layer for an Integrated Modular Avionics (IMA) System with ease," noted Neil Davidson, Principal Software Engineer, Avionic Systems. "We have used object oriented UML modeling tools on previous programs, however the Hawk program presented us with a new challenge of generating safety-critical SPARK Ada from a UML model."

"RtS' extensibility (both OLE automation and scripted code generation) enabled us to tailor the tool to meet our needs; we extended RtS so that we could enter SPARK annotations as part of the UML design model. We are able to manipulate SPARK annotations at the design level and then forward generate them as part of the source code, ready for SPARK Examination. We have gained multiple benefits from this: (a) freeing engineers to think about the design implications of SPARK; (b) eliminating the manual typing effort traditionally associated with SPARK; (c) automatically observing semantic rules for the annotations; and (d) ensuring compliance with our coding standards. These have already translated into a significant cost benefit on our project. RtS' extensibility has also allowed us to generate a semi-automated MIL-STD-498 design document from our UML model."

DDC-I - DDC-I Updates Sun/Solaris-Hosted TADS Development System

Date: Sat, 5 Jun 2004 16:12:59 +0200
Subject: Embedded News from DDC-I - DDC-I Online News
http://www.ddci.com/news_vol5num3.shtml

Phoenix, AZ March 1, 2004 DDC-I is pleased to announce the release of a fully updated Sun/Solaris®-hosted TADS Ada Development System (TADS). This release incorporates the full complement of improvements and enhancements integrated during the now-completed TADS for PC/Windows® rehosting development effort. After releasing TADS-1750A in November 2003, TADS-i960 and TADS-68xxx are scheduled for full release in March 2004.

"Offering our TADS customers consistent performance across the full range of hosts and targets is the best way to support changing needs. Whether they plan to rehost using a PC/Windows platform, perform regular program maintenance, port legacy code using existing Sun/Solaris resources or any combination of the above, all variants are now uniform" explains DDC-I Senior Software Engineer Michael Hash.

Specific enhancements include: improved compiler symbol management, enhanced internal consistency to facilitate program linking; enhanced and generalized serial communication connectivity of the AdaScope debugger (for better compatibility with standard serial port expander boards in the host computer); improved handling and resolution of internal compiler errors, as well as improved machine code insertion of 1750A assembly instructions into Ada source code.

A mature solution for each target, TADS offers classical, Ada specific and target specific compiler optimizations to deliver performance benefits tailored to processor architecture. TADS generates the most compact code available via highly optimizing compilation, selective linking and modularization of the run-time system.

"Many aerospace, avionics, defense and other customer programs still depend on TADS safety-critical real-time embedded system development," Hash concludes," and DDC-I's philosophy of #1 in Customer Care focuses on providing the best possible software tools for every client, coupled with superior engineering services and customer support."

DDC-I - DDC-I Releases Updated Version of Proven DACS IDE

Date: Sat, 5 Jun 2004 16:12:59 +0200
Subject: Embedded News from DDC-I - DDC-I Online News
http://www.ddci.com/news_vol5num3.shtml

Phoenix, AZ Feb 25, 2004 Maintaining the high performance standards of the established DDC-I Ada Compiler System (DACS) DDC-I today announced the release of Windows-hosted DACS-PC version 4.7.16 for both DACS-Native and DACS-MAPP variants.

"Responding conscientiously to customer needs has always been a top priority at DDC-I, and this comprehensive new release made originally for the Boeing/Sikorsky Comanche team can now deliver the latest improvements to the IDE made for specific customers to every DACS user," explains DDC-I DACS Product Champion Richard Frost.

Hosted on PCs running Windows NT and targeting Windows NT and embedded 80x86 cross-targets, the underlying Ada tool set is based on the DACS-80x86 cross development system in use for over 10 years. Fully tested and QAd, the release comprises customer-requested changes and fixes included in general release 4.7.15b and specific patch releases bundled with upgrades and enhancements to the GUI, AID tool, compiler and target linker.

Generating native Windows NT console applications and embedded applications for Intel Real Mode 186, Flat Mode (386, 486, 586) and Protected Mode (2/3/4/586) from the same integrated environment, the environment maintains a constant appearance regardless of selected target. Users can easily prototype applications using the native system and rebuild them for the cross target when they are stable.

Specific improvements in v4.7.16 include; an XML output option in the Ada Information Dumper, offering customers an easier format to parse for support tools; addition of NVM functionality to the burn tool, alongside current SUROM capability; the ability for DACS-MAPP and DACS-FM586 to co-exist in single application development situations.

"Our main mission remains providing a fully integrated user environment that flawlessly performs every common software development task, from simple text editing to compiling, linking, executing, debugging and managing the program library, as well as project and user-level tool customization options and project management support," Frost concludes.

DDC-I SCORE® Product Line Support for Dy 4 Systems SVME/DMV-181 Single Board Computer

Date: Sat, 5 Jun 2004 15:59:41 +0200
Subject: Embedded News from DDC-I - DDC-I Online News
http://www.ddci.com/news_vol5num5.shtml

Phoenix, Arizona, 20 April 2004 DDC-I announced today the availability of a support package enabling the versatile SCORE® integrated development environment (IDE) system to target and debug applications for Dy 4 Systems popular PowerPC-based SVME/DMV-181 single board computer (SBC).

"Packed with features to satisfy systems integrators real-world requirements, a high level of integration and diverse I/O capabilities make Dy 4s 181 a target platform of choice," explains David Mosley, Engineering Manager and SCORE® Product Champion. "We created the 181 board support package for SCORE® to maximize the development options available to customers designing products for the latest generation of civilian and military embedded systems."

The SCORE® Multi-Language Debugger support is provided via the Abatron BDI2000 JTAG interface. Using this JTAG interface means that no debug monitor needs to be present on the board to support debugging. The application code in its final form can be easily debugged over Ethernet from the host-based debugger to the Abatron JTAG device.

The first IDE with multi-language, multi-target and multi-host capabilities based on non-proprietary open system standards, SCORE® meets an increasing need to combine reusable software components, often written in different languages, targeting different microprocessors and created on different platforms. The front-end incorporates project management tools, online help, tool activation, numerous efficiency features and a universal interface for compilers and tools, using open standards for easy third-party product integration.

Known in the defense and aerospace community for leading-edge ruggedized products, Dy 4 Systems SVME/DMV-181 SBC combines a processing core based on the powerful AltiVec-equipped PowerPC 7410 processor with an unmatched I/O complement. Two independent 10/100 Ethernet ports provide redundancy and survivability in a networked system environment, alongside two PMC mezzanine I/O expansion sites, SCSI, six serial ports and two USB ports.

For military and aerospace development requiring stringent DO-178B certification, both DDC-I's SCORE® IDE and Dy 4's SVME/DMV-181 provide full DO-178B capability.

"Dy 4 has seen great interest from customers in using the SVME/DMV-181 for applications requiring high-reliability," says Mosley. Dy 4 works with leading OS and software tools vendors, "and we have worked closely with them to ensure SCORE® will support many of the advanced features of their most popular single board computer."

DDC-I - Phoenix-based DDC-I Unveils Atlas IT Consulting

Date: Sat, 5 Jun 2004 15:52:32 +0200
Subject: Embedded News from DDC-I - DDC-I Online News
http://www.ddci.com/news_vol5num4.shtml

Experienced software developer now delivers affordable IT consultancy and services for small-to-medium Phoenix-area business clients

In the 21st Century every company requires IT infrastructure to plug into the wired world of modern business, but even a small scale, single-site network can be more than a full-time job to manage. Whether your company needs help navigating the latest security concerns, developing improved network reliability, or straight advice about configuration and storage, only one Phoenix-area IT company can deliver a truly high-flying global experience to the greater Phoenix business community.

"Outsourcing your business IT challenges to an experienced company can take significant risk out of developing and managing your networks and infrastructure," explains Ole Oest, President of parent company DDC-I, a Phoenix-based technology company that for over twenty years has provided software and development services to a veritable "who's who" of large industrial clients such as General Dynamics & Honeywell. "With Atlas IT Consulting we can maximize productivity for your company by making the same talented team working for DDC-I available to solve your current and future IT challenges. Our goal for you increased profits."

According to Oest, while consultancy is primarily used to provide support for the permanent IT contingent in larger organizations, they can also serve as an extremely effective remedy to the problems and concerns common at small-to-midsize organizations with fewer resources than the larger firms. In addition to relieving the HR department of recruiting, screening, and evaluating IT talent, intelligently outsourcing specific IT design, management, support and upgrade projects can reduce the strain on a limited IT budget.

With a specific focus on high quality customer care, Atlas IT offers one point of contact and a commitment to keep systems running reliably. Additionally, Atlas IT can handle virtually any situation facing your company and IT infrastructure. Whether the project is analysis and needs evaluation for new systems, a major systems integration, remote monitoring applications, net connectivity and security, proxy configuration, on-the-spot troubleshooting or project management from inception to deployment, Atlas IT Consulting will provide knowledge and support.

Atlas IT Consulting offers big business experience to small businesses of all kinds. "The creativity, engineering support and customer service that helped Boeing launch the 777 is what were now offering to every small-to-medium business in the greater Phoenix area," Oest concludes. "Our two decades of real-world experience in aerospace and safety-critical software applications, where system failure is simply not an option, gives us the tools we need to help every client excel."

ExcelSoftware - WinA&D & WinTranslator

From: Tools <excel@lobo.net>
Date: Wed, 03 Mar 2004 15:48:33 -0700
Subject: UML for Ada
Newsgroups: comp.lang.ada

Those interested in UML modeling, code generation and reengineering Ada code to models can find information at: http://www.excelsoftware.com/umlforada.html

[Form the web page: WinA&D is a comprehensive tool for doing system analysis, requirements specification, software design, code generation and custom reports. It supports a broad range of methods and notations including structured analysis and design, UML, data modeling and real-time, multi-task design. Version 3.5 adds advanced capabilities for modeling and generating Ada 95 code. WinTranslator generates diagrams and dictionary information from existing source code. Generate data models from SQL, UML class models from C++, Java, Delphi and Ada

95 or structure charts from C, Pascal, Basic and Fortran. Version 2.2 generates rich UML models in WinA&D from Ada 95 code. Version 2.3 adds sophisticated comment handling features. -- su]

McKae Technologies - DTraq

From: Marc A. Criley <mc@mckae.com>
Date: Tue, 02 Mar 2004 23:10:47 GMT
Subject: Announce: DTraq 0.980 is now available
Newsgroups: comp.lang.ada

[See also "McKae Technologies - First Public Release of DTraq Data Logging and Playback Debugging Tool" in AUJ 24.4 (Mar 2004), p.213. -- su]

McKae Technologies announces the release of DTraq 0.980, an Ada 95 data logging and review tool.

DTraq is a data logging and playback debugging tool providing near realtime data logging and analysis to aid debugging and validation. Captured, or 'tapped' data from a program can be viewed live while the program is running or, since it is being logged to a file, played back or printed out later for off-line review and analysis.

DTraq differs from other logging and playback tools in that no data layout maps or byte interpretations or "data dumpers" need to be manually created. Nor is the application responsible for converting the raw binary data to text form before logging it. DTraq handles all conversion automatically by scanning the application's source code, identifying tapped data items, and extracting the information it needs to properly convert and display the logged items-simple scalar items as well as arrays and records. When the layout of data items change, rescanning automatically picks up the changes.

DTraq requires GNAT 3.15p due to its reliance on the Ada Semantic Interface Specification (ASIS) and has been validated on Red Hat 9 Linux.

Source and executables are available on the DTraq home page:

http://www.mckae.com/dtraq.html, along with the comprehensive and up-to-date user manual -- www.mckae.com/dtq_common/DTraq.pdf.

Enjoy!

Marc A. Criley, McKae Technologies, mc@mckae.com

Ada and Linux

Ada and Linux 2.6

From: Mark H Johnson <Mark_H_Johnson@raytheon.com>
Date: Fri, 09 Apr 2004 11:53:56 -0500
Organization: Raytheon Company
Subject: Re: Ada compiler
Newsgroups: comp.lang.ada

Sergey wrote:

> Guys, I'm experienced C++ programmer and pretty new to Ada. Despite from that I'd like to use Ada for my current project (it's not a big deal but ...) and I have same questions for you ... I've to use RHEL-3 (NPTL is the keyword here) - a platform where (as I heard) gnat-3.15p doesn't do well (tasking is completely broken). RHEL-3 comes with gcc-3.2.3 and contains gnat as well, but (been googling a little) I've found that it isn't recommended for use in production ... So my question - what Ada compiler are you guys using on RHEL (or RHL-9)?

You have a few alternatives:

- if you care about "use for production", you should probably get a support contract from Ada Core Technologies. That way, you get a validated compiler built for RHEL. ACT's support is a little expensive and geared for large development teams but the quality of support has been excellent.

- if you don't have $$, you can still use 3.15p if you specify

  export LD_ASSUME_KERNEL=2.4.1

to get the old tasking behavior (which works fine). This was described on comp.lang.ada some time ago.

- or you can try gcc-3.2.3. Perhaps someone else can comment on how well it works (or doesn't).

Debian policy for Ada

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 03 Apr 2004 14:08:18 +0200
Subject: RFC: Ada policy for Debian
Newsgroups: comp.lang.ada

I have written a short document that formalises my "Ada Policy for Debian"; that is, how I think packages should be laid out. I have posted it on: http://users.skynet.be/ludovic.brenta/ada-policy.html

This is a request for comments. Please reply to me privately with your thoughts on this document and I can amend it as necessary. My intention is to eventually include it in Debian; either as documentation with package `gnat', or as a separate package.

X11 Bindings

From: Ben Atkin <bma3@dana.ucc.nau.edu>
Date: 16 Mar 2004 13:20:25 -0800
Subject: Xlib Binding or Re-implementation?
Newsgroups: comp.lang.ada

Do any of you know of any good, free Xlib bindings? I have run into a lot of dead links.

Second, if I create my own, do you think it would be better to:

A. Write a thin binding to the C Xlib, using pragmas.

B. Write a thick binding to the C Xlib (don't know how I would go about this).

C. Port Xlib.c, Xlib.h, etc. to Ada, using sockets. Then I could make some changes that would make it more robust (the same things that the freedesktop.org people are trying to do with http://www.freedesktop.org/Software/xcb, but using Ada, which goes really well with all of their suggestions).

I realize this could be a long project, but I am so dissatisfied with GTK's approach that I think a new approach is needed. Besides, more experimental projects are needed on the 'nix desktop. Copying is a terrible way to develop software.

Thanks for any advice!

From: Jacob Sparre Andersen <sparre@nbi.dk>
Date: 01 Mar 2004 23:16:38 +0100
Subject: Re: X11 binding
Newsgroups: comp.lang.ada

The file <http://dk.gnuada.org/linux/rpms/3.13p/glibc2.1/SRPMS/x11ada-1.30-13.src.rpm> is likely to contain the source code for an Ada binding to raw X11.

You might be interested in using something like GtkAda instead. GtkAda is a thick Ada binding to both X11 and the Microsoft GUI API (and probably also other GUI API's).

From: Jeff C <jcreem@yahoo.com>
Date: Tue, 02 Mar 2004 00:09:14
Subject: Re: X11 binding
Newsgroups: comp.lang.ada

There is one that is not a broken link at http://www.adapower.com/lab/adax.html

But I agree with the other poster. I recommend you go with gtkada instead: gtkada.eu.org

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 17 Mar 2004 11:34:25 +0100
Subject: Re: Xlib Binding or Re-implementation?
Newsgroups: comp.lang.ada

There is a binding to Xlib, Xt, and Xm (Motif/Lesstif) called AdaBindX, at http://home.arcor.de/hfvogt/programming.html. It does not have a binding to Xaw, the Athena widget set. I am packaging it right now for Debian, it should be uploaded later today.

Maintaining AdaCL in Debian

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 18 Mar 2004 01:19:51 +0100

Subject: Re: Xlib Binding or Re-implementation?
Newsgroups: comp.lang.ada

Ben Atkin wrote:

> > What is it that you don't like about Gtk?

> It isn't Gtk in particular. It's that GUI libraries for 'nix, other than Qt, don't encourage good GUI design principles. In fact, if you compile their examples on GTK's website, you will get a broken GUI. By broken I mean that pressing Esc doesn't close a dialog box, and pressing Enter in a single-line text field doesn't cause the default button to be pressed and the appropriate action to be taken.

That is true, but at least GNOME has published human interface design guidelines which, if followed, lead to this consistency. Xlib doesn't have such guidelines, it's not even a toolkit. If you mean to use Xaw (the Athena widgets) or Xm (Motif or Lesstif widgets), you won't necessarily have a good GUI either.

> Plus, GTK programs are full of other types of bugs, many of which could be prevented by using something other than type-unsafe macros, leading to spaghetti code.

This is comp.lang.ada; we do not use type-unsafe macros here :)

The designers of GTK+ never intended for GTK+ applications to be written in C. I personally attended a presentation by Owen Taylor, a member of the core GTK+ team, at FOSDEM 2003. He specifically said "Do Not Program In C", and he explained that his employer Red Hat uses Python for all their GUIs.

The main (perhaps only) reason why GTK+ itself is written in C is to make it easy to write thick bindings to it from high level languages (see http://www.gtk.org/bindings). GtkAda is one such binding, and a pretty good one at that.

> Part of me wants to jump on the GTK bandwagon, and help things progress toward a standard. But what I really want to progress toward is having well-designed GUI's.

Over and above the thick binding to GTK+, GtkAda provides several rich widgets (canvas and MDI come to mind) and certainly the authors would gladly accept your contributions.

There is also Glade, the graphical UI designer, which can generate Ada source code. If you're really interested in good UI design, you'll probably be much more productive with it than with Athena or Lesstif.

I suggest you really have a good look at GtkAda; you will probably find that it already provides much more of what you want than do Xaw or Xm. And if something is missing, please offer to contribute :)

(You may also be interested in GNUStep, which is one pretty good toolkit with UI guidelines and a graphical designer, but for that you'd have to go to the Objective-C newsgroup. GNUStep has been quite active in recent months.)

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 18 Mar 2004 14:36:52 -0600
Subject: Re: Xlib Binding or Re-implementation?
Newsgroups: comp.lang.ada

Preben Randhol wrote:

> I would rather have Gwindow or Claw ported to work on Linux and Mac OS, than a new library. I don't want to put you off what

I have no objection to a Mac Claw. It's not clear to me if the interface of Claw is too Windows-specific. We had considered a system-independent layer that would be built on top of classic (Windows) Claw, but to date we haven't bothered because the need seems to be met by GTKAda. But if anyone wants to persue it, I'd be happy to hear from them.

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 18 Mar 2004 11:22:41 +0100
Subject: Booch Components and AdaCL in Debian
Newsgroups: comp.lang.ada

I have looked at the AdaCL project on SourceForge, and I have to say I am quite impressed with the activity that takes place there. However, the pace of new releases is also a concern to me; I already maintain 17 source packages and it would be difficult for me to package a new version of AdaCL every couple of weeks. I just won't have enough time. In addition, if the binary interface to the library changes too often, people will be reluctant to use the library in their programs. The binary interface normally changes at every major release; this is reflected in the soname of the shared library.

I can see two solutions to this problem of too frequent releases:

- let somebody else maintain AdaCL in Debian; this includes not only packaging each new version, but also tracking bugs, being the front-line of support for Debian users, and interacting with the upstream authors on SourceForge. I could provide assistance to get going. In fact, generally speaking, I would really like it if someone else stepped up to package more Ada software in Debian.

- slow down the pace of new releases to, say, one every 2 months. This implies no bug-fix releases (unless absolutely required, of course). This implies an automated test suite to ensure top quality of each release (hint: AUnit is already in Debian).

I don't think it is desirable to skip some versions of AdaCL in Debian, because newer versions seem to provide valuable bug fixes.

KDE Binding

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Wed, 31 Mar 2004 15:14:41 +0200
Subject: Re: KDE Anyone?
Newsgroups: comp.lang.ada

Chris wrote:

> Does anyone know of a KDE binding for Ada. I looked on the net and found someone had started one. Where's it at? Is it active? The post talked about having a fairly substantial binding to QT, but it was July 2003. What's happened since then?

Well there is QtAda 95: http://sourceforge.net/projects/qtada

But the project seems dead.

> How hard is it to bind to C++? I gather objects complicate things a lot. Would using a C++ based binding on Linux force me to move to something like Gcc 3.xx based compilers. This is not really a problem (I like playing with experimental sw!), it's simply that AFAIK there's no firm schedule on when the Ada compiler in Gcc 3.xx will be stable. Like I say not a problem, just something I need to account for.

I don't know. There are quite a few pragmas for c++ support but I have not found and docu or examples which clearly state how to use them. One could try the make a binding to QString and see how it goes.

From: Jeff <jeff.huter@bigfoot.com>
Date: 31 Mar 2004 15:55:28 -0800
Subject: Re: KDE Anyone?
Newsgroups: comp.lang.ada

If you want to do a binding, then you may want to check out the C binding at qtcsharp.sourceforge.net. QT# is actually based upon.

From: Jeff <jeff.huter@bigfoot.com>
Date: 30 Mar 2004 09:33:28 -0800
Subject: Re: KDE Anyone?
Newsgroups: comp.lang.ada

One possibility may be to use A# with QT# and mono. Info about A# is located at http://www.usafa.af.mil/dfcs/bios/mcc_html/a_sharp.html. However, A# appears to target only Windows. I have no idea how hard it would be to port A# to *nix and mono. But, it may be easier than creating a new binding for QT.

From: Oliver Kellogg <kellogg@freenet.de>
Date: 12 Apr 2004 12:52:14 -0700
Subject: Re: KDE Anyone?
Newsgroups: comp.lang.ada

BTW, it looks like kalyputs and smoke have superceded qtc, see: http://webcvs.kde.org/cgi-bin/cvsweb.cgi/kdebindings/kalyptus/

http://webcvs.kde.org/cgi-bin/cvsweb.cgi/kdebindings/smoke/

Auto_Text_IO 3.02

From: Stephen Leake <stephen_leake@acm.org>
Date: 05 Apr 2004 21:37:08
Subject: Auto_Text_IO 3.02 released
Newsgroups: comp.lang.ada

Auto_Text_IO 3.02 released

This is just a bug fix release.

Auto_Text_IO is a tool for automatically generating a Text_IO package for an Ada package. The Text_IO package contains Put and Get subprograms for all the types in the Ada package, based on Ada.Text_IO. The Put and Get subprograms use named notation aggregates (although Get does not support the full flexibility of Ada source code). This makes it extremely easy to write readable unit tests for the Ada packages, and to provide persistent storage in a readable form.

See:
http://www.toadmail.com/~ada_wizard/ada/auto_text_io.html
for more info.

From: Jeff C <jcreem@yahoo.com>
Date: Fri, 13 Feb 2004 03:40:29 GMT
Subject: Re: ELF Header Symbol Extraction
Newsgroups: comp.lang.ada

James Amor wrote:

> Does anyone know of an Ada library that extracts ELF header information from a binary? Anyone got any suggestions on how to do it if I can't find a library?

How about something like:
http://savannah.nongnu.org/projects/bfdada/
(Not really sure of its status)

Ada and Microsoft

Microsoft Fundation Classes

From: Steve <steved94@comcast.net>
Date: Mon, 15 Mar 2004 02:48:07
Subject: Re: MFC & Ada
Newsgroups: comp.lang.ada

Szymon Guz wrote:

> does anyone use MFC.dll in Ada programs?

AFC is a thin binding to MFC:
http://www.jswalker.demon.co.uk/jswtech.htm#AFC

I haven't used the binding, but from what I understand it is rather complete.

From: Szymon Guz <guzo@stud.ics.p.lodz.pl>
Date: Wed, 17 Mar 2004 18:40:28 +0100
Subject: Re: MFC & Ada
Newsgroups: comp.lang.ada

Randy Brukardt wrote:

> I would have to ask why you'd want to. The reason that libraries like Claw and GWindows exist is so that you can build Windows applications in Ada, with an Ada mindset. You're much better off using one of those libraries than MFC with an Ada program, and they cover pretty much the same ground. (And Claw at least works with all of the Ada compilers for Windows, so you don't have to be locked into a specific compiler.)

Yea, I know all that but I have to use the mfc.dll in Ada programs for my masters thesis.

References to Publications

SPARK cited in NCSP Software Security Report

From: Rod Chapman <rod.chapman@praxis-cs.co.uk>
Date: 5 Apr 2004 08:00:55 -0700
Subject: Praxis and SPARK cited in NCSP Software Security Report
Newsgroups: comp.lang.ada

Those of you interested in Ada-related ammunition and news might be interested in the recent National Cyber Security Partnership report "Security Across the Software Development Life Cycle" here: http://www.cyberpartnership.org/init-soft.html

The report cites Praxis Critical Systems' "Correctness by Construction" (of which Ada and SPARK are a big part) software process as one of very few that can deliver the quality and defect rate needed for present and future secure systems.

The Appendix to the report also includes our full "Correctness by Construction" paper, and metrics for various projects, 4 of which (all except CDIS) are (or include) SPARK to some extent.

All the best,

Rod Chapman, SPARK Team, Praxis Critical Systems

DDC-I Online News

From: jc <jcdk@ddci.com>
Date: Wed, 3 Mar 2004 15:52:00 -0700
Subject: Real-Time Industry Updates - News from DDC-I
To: 2D March 2004 Online News DK <jcdk@ddci.com>

DDC-I Online News - March 2004, Volume 5, Number 3 - [http://www.ddci.com/news_vol5num3.shtml] A monthly news update dedicated to DDC-I customers & registered subscribers.

TADS Upgrade Now Available. Offers Consistent Performance Across Full Range of Hosts & Targets for TADS Customers

Updated Version of Proven DACS IDE. Version 4.7.16 Now Available for Both DACS-Native and DACS-MAPP

Thoughts from Thorkil. Unsigned Numbers in Ada83 (1st of 2 segments)

A Few More Patterns. For "Introducing" Change

From: jc <jcdk@ddci.com>
Date: Tue, 4 May 2004 13:53:39 -0700
Organization: DDC-I
Subject: Real-Time Industry Updates - News from DDC-I
To: 4D May 2004 Online News DK <jcdk@ddci.com>

DDC-I Online News -May 2004, Volume 5, Number 5. [http://www.ddci.com/news_vol5num5.shtml] A monthly news update dedicated to DDC-I customers & registered subscribers.

SCORE® now supports Dy4 SVME/DMV-181. Supports many of the advanced features of their most popular single board computer.

Thoughts from Thorkil - Date, Time and Ada (1). The concept of local time can pose challenges for using dates.

The Real Power of Retrospection. Read how an entire company, not just one team, can experience this.

Ada 95 Books

From: Preben Randhol <randhol@pvv.org>
Date: Thu, 5 Feb 2004 15:09:55 +0000
Subject: Re: Good book for Ada 95 wanted
Newsgroups: comp.lang.ada

> Can anyone recommend a book on Ada 95? I found a lot books but which one is a good one for starting? I have a lot programming experinence with Java or C/C++.

Of the on-line books (see http://www.adaworld.com/ or http://www.adapower.com/):

Ada Distilled

Ada 95: The Craft of Object-Oriented Programming

Object Oriented Programming in Ada 95

Of the books that you have to buy or borrow at your library:

Ada as a Second Language (2nd Edition), Norman Cohen. McGraw Hill, 1996. (ISBN 0-07-011607-5)

Programming in Ada 95 (2nd edition), Joh

n Barnes. Addison-Wesley, 1998. (ISBN 0-201-34293-6)

Ada 95 for C and C++ Programmers, Simon K. Johnston. Addison Wesley, 1997 (ISBN 0-201-40363-3)

I would recommend Ada as a Second Language (2nd Edition) or Programming in Ada 95 (2nd edition).

Ada Inside

ACT develops ejector seat system for pilots

From: AdaIC Technical Webmaster <webmaster@adaic.com>
Date: Thu, 15 Apr 2004 22:22:26 -0500
Subject: [AdaIC] Ada in use: Ejector seats
To: <announce@adaic.com>

The Navy has deployed a new ejector seat system for pilots, developed in Ada by Ada Core Technologies. Read all about it in "Punching Out!" in Military Aerospace Technology at http://www.military-aerospace-technology.com/articles.cfm?DocID=411

As the author notes, "pilot safety is paramount [when] the ejection seat becomes the first stage in a multi-layered approach to bringing [pilots] down to earth. The newest system being deployed will save lives and money."

Indirect Information on Ada Usage

Date: Sun, 15 Feb 2004 09:06:10 +0100
Subject: BE-Brussels-belgium-Ada Programmer with C
X-URL: http://job.monster.be/

My client is currently searching for an Analyst Programmer with Ada and C Pramming skills for a critical project for full life cycle development. A successful candidate will have at least 3 years experience in:

Profile: Ada + C Programming Expert

Duties: Collecting user requirements. Analysis of requirements and deriving software requirements. Updating Development docs. Modification of Ada and C Code. Writing Test Plans. Executing Tests. Writing Test Reports

Human Skills: Social, Ability to work in mulicultural environment, Proactive and entrepreneurial, Team worker, Quality and Safety minded

Job Requirements: OO, Ada83, Ada95 and ANSI C, Configuration Mgmt tools, Data Modelling Techniques.

From: Nick Riccione <nick@btstaffing.com>
Date: Mon, 22 Mar 2004 15:55:30 -0600
Subject: Ada Job Opportunities
To: team-ada@acm.org

I have a client that has 4 openings for Ada Software Engineers in the Dallas, TX area [...]

Job Description:

* The job will include development of new, and modification of existing, system software and support software components.
* Software engineers will work through the entire software development life cycle from requirements analysis and through integration, test and delivery.
* An object-oriented based, open architecture with integrated COTS applications are applied in the development of new system software.
* Opportunities exist in several domains from device drivers, network communications, infrastructure, application development, tools, simulations, and GUI.

Required Skills:

* Bachelors degree in Electrical Engineering, Computer Engineering, Computer Science,
* Software development experience in several of the following areas is preferred: Ada, Ada-95, Unix, C++, C, GUI design, UML, and Object Oriented Methodology.
* Knowledge of Solaris architecture and near-real-time asynchronous operations is desirable for most positions.

Ada in Context

Software Quality Measurement

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Tue, 10 Feb 2004 10:40:40 +0100
Organization: Adalog
Subject: Re: Software Quality & Fault Measurement
Newsgroups: comp.lang.c++,comp.lang.c,comp.lang.ada,comp.lang.java.advocacy,comp.lang.perl.misc

Michael Kelly wrote:

> I'm trying to locate information on quality measurement studies that have been done on projects using C, C++, Ada, Java, and Perl. I'm particularly interested in materials regarding fault measurements. Any pointers would be greatly appreciated.

A famous paper on C vs. Ada is "Comparing Development Costs of C and Ada": http://www.adaic.com/whyada/ada-vs-c/cada_art.html

From: Peter Amey <peter.amey@praxis-cs.co.uk>
Date: Thu, 12 Feb 2004 10:08:34 +0000
Subject: Re: Software Quality & Fault Measurement
Newsgroups: comp.lang.ada

Although it wasn't primarily a comparative study, some useful evidence emerged from the Lockheed C130J programme and, in particular, the certification effort sponsored by the UK MoD on that aircraft. Relevant papers from Crosstalk Journal include:

"Correctness by Construction, Better Can Also be Cheaper" http://www.sparkada.com/downloads/Mar2002Amey.pdf

and

"Software Static Code Analysis Lessons Learned" by Andy German, QinetiQ Boscombe Down. DoD CrossTalk Journal, November 2003. http://www.stsc.hill.af.mil/crosstalk/2003/11/index.html

Executive summary: C130J (Hercules II) code, already cleared to DO178 levels A or B, in a variety of languages from a variety of vendors, was reviewed and examined by a MoD-appointed contractor. Residual error rates for Ada were about an order of magnitude smaller that for C-based languages. SPARK had significantly lower error rates than Ada.

Development over Symbian OS

From: Alexander E. Kopilovich <aek@VB1162.spb.edu>
Date: Fri, 14 May 2004 00:37:00
Subject: Symbian OS (was: Re: Ada used in General Aviation (GA) applications?)
Newsgroups: comp.lang.ada

Martin Dowie wrote:

> A "Dummies Guide to porting GNAT" covering both processor and OS would be a huge help. I'd love to have Ada available for SymbianOS (or it's predecessor EPOC32).

Being more or less familiar with Symbian OS (mostly SonyEricsson's flavour - Symbian 7.0, much less Nokia's Series 60 flavour - Symbian 6.1) I think that any guide of porting GNAT will not bring you near to that aim.

Actually, current Symbian C++ development toolset consist of C++ compiler (usually Metrowerks CodeWarrior is used for Symbian 7.0 and MSVC 6.0 for Symbian 6.1, although there are SDKs for other compilers, for example, Borland's C++ Builder), resource compiler, linker (with addition for resources) - all that produces an executable for emulator only - and emulator itself. Production version of the program (which will be loaded into smartphone) has to be compiled and built with another toolset, which is GCC-based.

So, in principle one may try to port GNAT for Symbian, but there will be severe haedaches with GCC versions, and you'll not be able to use the emulator for development.

Then, even if you adapt GNAT for Windows for use with the emulator, you still be quite restricted, because the emulator has problems with support of tasking - in Symbian terminology it supports "threads", but not "processes". Well, C++ development on emulator has the same limitation, so I just warn you that you should not expect too much from Ada tasking on emulator (if you aren't going to develop better emulator).

But all that is less than half of work, I think - because there are mountains of APIs for Symbian (and many of them differ significantly between Symbian's flavours). And it will be quite serious work to provide Ada bindinds even for most needed APIs. (As you can expect, those APIs/classes/etc. are only partially documented, and the source code is not available after EPOC32 R5).

Actually I thought that RR Software's Janus/Ada might be more proper thing (than GNAT) for targetting at Symbian, and several times I tempted to tell Randy about this opportunity... but there is a lot of work, and perhaps small chances that that work will be compensated by sales. If there were interest and some funding from one of smartphone vendors that use or plan to use Symbian (Nokia, SonyEricsson, Siemens, Samsung, etc. ... I'm not sure about Motorola) - it will be another matter, but they still are trying to push Java (without much success, though - as far as I can see).

Naming Convention for Ada Classes

From: Peter C. Chapin <pchapin@sover.net>
Subject: Naming convention for classes?
Date: Tue, 03 Feb 2004 23:52:28
Newsgroups: comp.lang.ada

I'm a C++ person learning my way around Ada. Naturally I tend to think about things in a C++ way and that leads me to various questions. I notice, for example, that there seems to be two (at least) somewhat different ways to name classes in Ada. The first way gives the class name to a package producing the following effect.

```
package Date is
 type Object is private;
 procedure Advance
 (Item : in out Object;
  Step : in Integer);
    ...
end Date;
```

The I can create Date objects by first withing Date and then doing

```
   Today : Date.Object;
   ...
   Date.Advance (Today, 100);
```

Another approach would be to give the type itself the "nice" name and regard the package name as "noise" used to avoid name clashes in large programs. For example:

```
package My_Library is
 type Date is private;
 procedure Advance
  (Item : in out Date;
   Step : in Integer);
   ...
end My_Library;
```

Then I might with My_Library and use My_Library and do:

```
   Today : Date;
   ...
   Advance(Today, 100);
```

The second approach seems natural but I notice that the Charles component library uses the first approach and it seems to work well in that case. For example, in my program I do:

```
package Vector is new
 Charles.Vectors.Unbounded
  (Index_Type   => Natural,
   Element_Type => Storage_Type);
```

Then I declare things to be Vector.Container_Type and use procedures like Vector.Append and Vector.Insert, etc. Now I'm building my own abstract type and I find myself waffling... should I use my intended name for the package or for a type inside the package? Is there some kind of "best practice" for this? Or am I off in the weeds here?

From: Jeffrey Carter <jrcarter@acm.org>
Date: Wed, 04 Feb 2004 00:27:18 GMT
Subject: Re: Naming convention for classes?
Newsgroups: comp.lang.ada

Some people are very attached to one approach or the other.

The two approaches you mention are called "use unfriendly" and "use friendly", respectively. There are arguments for and against both approaches. Ada's standard packages are use friendly. For example, package Ada.Strings.Unbounded declares type Unbounded_String.

If you create use friendly packages, you should not consider the package name to be noise. "Ada.Strings.Unbounded" is not just noise.

One approach is to use appropriate suffixes to create useful names:

```
package Date_Handling is
 type Date_Info is private;
 procedure Advance
  (Date : in out Date_Info;
   Num_Days : in Positive);
```

Note that with this approach the reader can tell that you're advancing a date by a number of days without any comments. Note also that negative numbers of days are not allowed, as that would not be considered advancing a date.

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Wed, 04 Feb 2004 15:13:03 +0100
Subject: Re: Naming convention for classes?
Newsgroups: comp.lang.ada

Peter C. Chapin wrote:

> > Now, this is not the only possible design pattern. For example, you can have the equivalent of "friends" in C++ by declaring two tagged types in the same package. The previous notation is no more applicable to this pattern.

> Yes, I see that. Coupling two classes by defining them in the same package provides some interesting options.

You should also remember that in Ada the concept of "protected" is implemented by child packages not child classes.

From: Georg Bauhaus <sb463ba@l1-hrz.uni-duisburg.de>
Date: Wed, 4 Feb 2004 14:57:02 +0000
Subject: Re: Naming convention for classes?
Newsgroups: comp.lang.ada

Unless I have missed it there is another style which uses plural for packages and singular for types. Thus you can have:

```
package Dates is
 type Date is ...;
```

and then:

```
 Dates.operation (today,  ...);
```

From: Jeffrey Carter <jrcarter@acm.org>
Date: Wed, 04 Feb 2004 19:01:28 GMT
Subject: Re: Naming convention for classes?
Newsgroups: comp.lang.ada

Peter C. Chapin wrote:

> Okay, so in other words either approach is considered acceptable by the community at large. It's a matter of style. At least that's the impression I get from your reply. Does that sound like a fair assessment?

To a large degree, yes, but as other replies have shown, there are cases where the use-unfriendly approach does not work.

From: Jean-Pierre Rosen <rosen@adalog.fr>
Date: Wed, 4 Feb 2004 09:57:06 +0100
Subject: Re: Naming convention for classes?
Newsgroups: comp.lang.ada

Since you said you were comming from a C++ background, let me explain some things you must understand first.

Ada has a so-called "building blocks" approach. Each feature provides a well-defined functionality, and the user builds the features he needs by assembling those building blocks. For example, packages provide encapsulation. Derived types provide support for inheritance. Tagged types provide support for dynamic dispatching.

Ada has no built-in concept of "Class" in the usual sense. If you consider that a class is an encapsulation with dynamic binding, then a class in Ada is a design pattern where you just declare one tagged type inside a package. *For this design pattern*, it makes a lot of sense to declare the package with the class name. A full discussion of this has been published in Ada Letters (Vol. XV, n°2): "A naming convention for classes in Ada 9X". The paper can be downloaded from http://www.adalog.fr/publica2.htm. Note that this convention works very well with "facets" generics, i.e. generics used to add properties to tagged types.

Now, this is not the only possible design pattern. For example, you can have the equivalent of "friends" in C++ by declaring two tagged types in the same package. The previous notation is no more applicable to this pattern.

For more sophisticated use of building blocks/design pattern approach, look at "Ada, Interfaces and the Listener Paradigm", a paper presented at Ada-Europe 2002 which is available from the same web page.

Ada Popularity

From: Carroll-Tech <andrew@carroll-tech.net>
Date: Sat, 7 Feb 2004 01:50:28 -0700
Subject: No call for it
Newsgroups: comp.lang.ada

> I've seen some fair sized 'C' and C++ projects. They are readable to 'C'/C++ programmers.
> I'd have loved to have used Ada, but there never was much call for it.

I've been reading posts to this newsgroup for some time now, as well as reading other information and I don't get the "never was much call for it" ideal about Ada. I'm not saying that anyone is wrong or right for saying or feeling that there "never was much call for it"; maybe there wasn't. I'm leaning more toward saying "it's a conscious choice".

I tell the students that I tutor that learning some Pascal or Ada would help them and they get scared. I mention doing a project in Ada and everyone looks at me like I'm out to punish myself. To me it isn't any easier to use C/C++, Java, Perl, Lisp or Prolog than it is to use Ada. How is it that Ada has this "super powerful", "super difficult", "there's not much call for it because it's too advanced and powerful" air about it when it's just another language? It's like saying that the machine code spit out of an Ada compiler has some mystical, magical properties that makes the Ada language more difficult to use.

To me, with Ada0Y coming out, the "not much call for it" attitude is the cliché to dismiss. Or at least one of the things to overcome.

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 07 Feb 2004 14:00:35 +0100
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada, comp.lang.c, comp.lang.c++, comp.lang.java

[In response to various articles for a specific poster:] I was thinking along the same lines last evening, and I came up with a small theory that explains why so few pople can be bothered to learn Ada. It goes like this: There are 3 types of languages.

The first type of language says "we're going to make programming easy". Of course, this is a lie, because programming is inherently difficult and no language can make it easy. These languages fake it by being simplistic. Java is the most prominent member of this family of languages; most scripting languages also fall in this category. Beginners tend to flock to these "easy" languages and never learn proper programming skills (like e.g. memory management. If some Java "guru" reads this, ask yourself this one question: how many threads does your program have, and please justify the existence of each thread).

The second type says "we will let you do anything, absolutely anything you want, and the power is in the hands of the True Programmers". Languages in this category include, among others, C and C++. Many people take a foolish pride in being called a True Programmer, and therefore like these languages. I myself once was in this category: I would show off my skills by writing a single-line program that nobody else could read. But humans write bugs, and these languages don't lend a hand finding these. Hence the famous buffer overflows.

The third type is what I would call the "zen master" type of languages. They treat you like an apprentice, slapping you on the hand each time you make a small mistake, and they scorn at you for choosing the quick and easy path -- which leads to the Dark Side. If you accept their teachings, you quickly become a Master yourself. If you rebel against them, you will never achieve Enlightenment and will always produce bugs. The "zen master" languages are Pascal, Modula, Oberon, and, master of masters, Ada. The beauty of these languages is that, once you are Enlightened, you can apply your wisdom to other languages as well -- but often would prefer not to.

From: Robert I. Eachus <rieachus@comcast.net>
Date: Sat, 07 Feb 2004 10:03:20 -0500
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada, comp.lang.c, comp.lang.c++, comp.lang.java

I think you are on the right track. When I am programming in Ada, I often spend most of a day coding. If I am exhausted at the end of it, I will put off compiling until the next day. Otherwise, I hand all the code to the compiler, and I am not surprised to be handed back dozens of error messages. Fix the syntax bugs, and now I get twices as many semantic errors. Kill all those and I am surprised if the test programs--often written between the package interface and the package bodies--don't run correctly.

For example, I recently finished writing a library of matrix operations which works with "views" that may be a submatrix of an existing matrix, and supports operations like Add(A,B) where the result is written in A. That's a bit tricky, but the real complex one is Mult(A,B) where the amount of temporary storage space for two N by N matricies is N. (A buffer that stores one row.)

Why am I mentioning this? After all the coding I had a bug in the Mult routine that the compiler didn't catch. A wrong subscript inside a loop. (Why am I doing this? To submit as a new benchmark for SPECfp. All that stuff is just scaffolding for implementing Strassen's algorithm efficiently.)

Since I don't take what the compiler tells me personally, I love the ratio of a hundred to one or so between compile errors and run-time bugs. Some people though look at a list of compiler error messages as if each one was a major failing on their part. Me? I could proofread the code carefully, but it is easier to let the compiler find out where I typed a comma for a period, and so on. And IMHO it would be nice if the compiler found all the typos, not just most of them. ;-)

Could I write the same code in C, C++, or Java? Sure. It is just much easier to let the Ada compiler do the heavy lifting part of the debugging, so I would still write in Ada, then modify the code to match the C, C++, or Java syntax. So to me, all that frequent hand-slapping is a major benefit.

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 08 Feb 2004 02:00:01 +0100
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada

> > Of course, this is a lie, because programming is inherently difficult and no language can make it easy.

> That's exactly what the assembly language programmers said about the first FORTRAN compiler, and it's equally wrong now. Sure, there are cases where you need to run DSP code and coordinate with the home base thirty million miles away using one space-hardened 386, and that's hard. Then there's the cases where you need two lines of shell to simplify moving files around, and that's something assembly or Fortran or Ada or Java would make much more complex then it is.

If you call this programming, then you're right. Scripting languages do have a place and purpose. I was more concerned with large-scale, real-world programming, where Ada shines but is being ignored by too many people. I was only trying to explain to myself why. I stand by my claim that no language can make programming easy. But a language help you find, or avoid, bugs.

> > (like e.g. memory management. If some Java "guru" reads this, ask yourself this one question: how many threads does your program have, and please justify the existence of each thread).

> In the Jargon file, there's a story of a man who bummed every cycle out of a poker program, even the initialization code, who spurned assembly language because it was too inefficient. How would you explain your choice of programming language to him?

I would ask him, "would you trust your own life to your program"?

> Who cares if there's a couple extra threads running? You make a big deal about languages that protect you against buffer overflows, why not use a language that protects you against memory leaks?

Because I want to control exactly how much memory my program uses, and I want to know exactly how many "a couple" means, and why these "couple" threads are necessary. You referred to embedded software for space-bound devices, this is one area where these questions are really important. I am willing to accept run-time inefficiency and "a couple extra threads" if justified. There are some languages that force these upon you and won't justify this cost.

> > The "zen master" languages are Pascal, Modula, Oberon, and, master of masters, Ada.

> Pascal is hardly usable, unless you use one of a dozen proprietary extensions. That's hardly "zen master".

That is true, but I meant "master" in the sense of "teacher". Pascal was quite good at teaching, and as a "master" it had quite a lot of apprentices.

From: James Rogers <jimmaureenrogers@att.net>
Date: Tue, 10 Feb 2004 02:45:27
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada, comp.lang.c, comp.lang.c++, comp.lang.java

> Thanks very much to everyone for the interesting info. It made me look more closely at Ada. It looks like it is indeed one of the safest languages among the ones that aren't garbage collected, which probably makes it suitable for programming things like airplanes, etc.:
> 1. hard real-time
> 2. bug-averse
> 3. not very performance demanding (don't know about other compilers, but they say GNAT produces slow executables)
> However, it does not look like it's a good match for me, since my needs are the exact opposite:
> 1. no real time
> 2. bugs welcome (but not wrong results) - lusers will not come near my programs
> 3. performance is highly important

I find your list of needs interesting.

How do you distinguish between bugs and wrong results? My experience is that bugs are detected because they produce incorrect results. If nothing goes wrong we do not declare the presence of a bug.

I think you will find, if you look into hard real-time systems, that performance is critical. While it is true that GNAT has produced relatively slow executables in the past, those same executables are often 3 to 5 times faster than early Java programs. I know that current JVMs have improved performance significantly. I speak of JVMs from around the year 2000. Other Ada compilers produce faster code than GNAT. Sometimes you get what you pay for. (GNAT is a free compiler in the GNU compiler chain).

What kind of performance measures do you use in your problem domain? C programmers are fond of fast code execution and fast compilation. C++ programmers have similar performance priorities, but are willing to sacrifice some compiler speed for the flexibility of templates. Java programmers frequently prize speed of coding, with the clever use of the large set of API libraries available to them. Ada programmers are fond of fast code and early detection of coding defects.

From: Ludovic Brenta <ludovic.brenta@insalien.org>
Date: 03 Apr 2004 21:46:14 +0200
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada

Marin David Condic wrote:

> Standards are a wonderful thing. Everyone should have one of their own. :-)

That's a nice way of describing Sun's policy :) I'm not as diplomatic as you are :)

> I'd differ in this respect: What makes something "The Best Tool For The Job"? Ada is superior in some technical aspects to other languages such as Java when considering the language definition alone. But if Ada doesn't provide as much stuff in its toolbox, isn't that in some respect making it a less satisfactory tool? Or if the implementation under consideration isn't very good, does it still qualify as the best tool just because in theory it could be better?

One of the nice things with Ada is that several implementations are available and can be evaluated. Surely, one of these implementations will turn out to be the best tool for the job. But, this requires effort to evaluate the compilers and libraries. In my experience, very few people actually make that effort. Do you think they evaluate the Java compiler and library? If they did, they wouldn't use them.

> The best tool for the job is the tool that lets me do the job while optimizing cost, schedule and quality. We tend to be convinced that Ada offers superior quality when one considers only the language proper. That may even be true in most cases, but it often ignores cost and schedule in evaluating "The Best Tool For The Job". "Quality" can

I differ in this respect. I find that I am much more productive with Ada than with Java, C++, C or Pascal. With Ada, I spend more time developing and less time debugging.

> be a relative thing - do I really need gold-plated screws when I'm building a birdhouse? Even if Java as a language doesn't detect as many bugs as does Ada, the presence of a well worn library means I'm not generating new and potentially buggy code to do the same thing. Might that not result in a higher quality end product - while reducing my costs and improving my schedule?

Ada has all the libraries needed to get that kind of leverage. The only problem is that these libraries don't come with the compiler, so you have to look for them. Or use Debian :)

> People don't select Java because they are fools. They often select Java over Ada for all sorts of legitimate and important reasons. If we want to get them selecting Ada over Java, we have to understand those reasons and come to the table being a better satisfier of those needs.

No, they are not fools. Their legitimate reason for choosing Java is that "everybody uses it", so it is easy for them to find disposable, cheap beginner programmers. They instruct these programmers to deliver buggy code quickly. These beginners are all too happy to use an "easy" and "fashionable" language, which their teachers at school taught them because "the industry demands it" (see the self-fulfilling prophecy there?). Then, when customers complain about the bugs, they blame the developers and fire them. Now is an "urgent" problem, so they quickly hire new apprentices and use them to churn out a new, buggy release that "fixes" the worst bugs while at the same time bringing new ones.

In short, their legitimate and important reasons are "job security for managers" and "repeat customers".

Development managers want a high price/quality ratio, i.e. a high development price and low quality. Why? Because, by spending lots of money on development, they are important. And by delivering low quality, they can say "See? My budget was too low!", and they can also charge big bucks to customers.

In corporations, customers want a high price, because spending lots of money makes them important. But they also want high quality. Once they've paid the big bucks for the expensive software they've selected, they are reluctant to admit that it's buggy or inadequate. Therefore, they pay even more for fixes and upgrades, and eveyone is happy.

Except for software developers and end users. The former work under pressure and get all the blame, while the latter have so totally lost confidence in the software that they blame it for everything.

From: Marius Amado Alves
<amado.alves@netcabo.pt>
Date: Thu, 8 Apr 2004 13:46:14 -0700
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada

Here's a idea to ease the adoption of Ada, and thus expand it, and thus augment the percentage of reliable software in the world, and throw some business our way along with it.

The main result is a CD+book that constitutes the big package everyone seems to be expecting, containing every resource/library required to learn Ada and build a vast class of applications, and easy to install and use.

The economical feasability of this project assumes that such a package does not exist already. Is GNAT Pro it? Is ACE? Another? If yes then stop reading here.

The realisation of this project requires money investment and/or resources, because I see no other way to do it than setting up a team of Ada library mantainers, application developers, authors, and perhaps trainers, holding at least one initial physical meeting in a laboratory somewhere. 10 or 20 people.

This requires coordination, leading to the selection of participants and identification of leaders. CEO+CTO is a likely structure. The very initial brainstorming could be done right here on CLA, but to advance it should rapidly shift to a dedicated structure. A virtual organization.

The first gathering would take a week or two and result in:

- the first prototype of the product
- a planned structure to produce and distribute copies of it
- coordination and maintainance structures strengthened.

The launch would of coincide with Ada 2005 :-)

This project clearly need managerial and commercial skills as well as technical, and as I said, money investment (e.g. for the meetings), and thus a business plan, and thus a market research. We have some market indicators from the people on this list, but perhaps not enough. […]

From: Jeffrey Carter <jrcarter@acm.org>
Date: Sun, 11 Apr 2004 01:01:17
Subject: Call for Ada
Newsgroups: comp.lang.ada

Wes Groleau wrote:

> Scenario: Two 100,000 SLOC collections.
> One in Ada, with (hope, hope) 1 error per 100 SLOC
> One in C, with 10 per 100 SLOC

At least 2 studies indicate that Ada reduces errors by a factor of 4.

> Assume an Ada programmer can find and fix a bug in a day, while the C guy takes two.

The same studies indicate that Ada reduces the time/cost to fix an error by a factor of 10.

> There are a hundred C hackers and 5 Ada hackers.

There are 100s of 1000s of C hackers. There are many Ada software engineers, but I'll accept 5 as the the number of Ada hackers :)

So, if Ada actually has 1 error per 100 LOC, then real numbers indicate 1000 errors in the Ada and 4000 in the C. If it actually takes 1 day to correct an error in the Ada, then real numbers indicate that it takes 10 days to correct an error in the C. So we get 1000 person-days/5 people = 200 days for Ada, and 40000 person-days/100 people = 400 days for C. Ada still wins.

I haven't seen any data to support it, but I suspect that C introduces more new errors for every error fixed than Ada, giving Ada an even greater edge, for the same reasons that C creates more errors in the 1st place.

From: Ludovic Brenta
<ludovic.brenta@insalien.org>
Date: 16 Apr 2004 13:00:13
Subject: Re: No call for it
Newsgroups: comp.lang.ada

You keep complaining that Ada lacks leverage, and you keep saying that Ada should this and Ada should that.

I think Ada does already provide a lot of leverage. From your posts, it looks like you are not aware of the existence of:

* AWS for web-based applications and web services
* GNADE for database apps (includes an Embedded-SQL preprocessor)
* GtkAda for portable GUI's, CLAWS and GWindows for Win32 GUI's
* XML/Ada for XML processing
* OpenToken for lexical analysis
* Charles or AI302 for data structures and algorithms

In terms of development tools, I keep being impressed with ASIS and the potential it has. At work I use an ASIS-based tool that generates code coverage information; I am learning how invaluable that tool is. I know of no other language that provides this, especially not in a _standard_ way.

You also say that students and hobbyists are not a good market to target. I disagree with this. Ada ia already entrenched in a very high-profile market - avionics, train control systems, nuclear industry - which is small, but sustains several healthy companies that provide development tools. In fact this small market is big enough to sustain comptetition between these vendors.

The problem is that the companies that make up this high-profile market are quite secretive, and do not normally advertise that they use Ada; they just do it with the understanding that Ada is a competitive weapon in their arsenal. This backlashes on them, as they now find it difficult to hire Ada programmers, or people willing to learn.

The only thing that Ada needs is visibility to the masses. So, if you really mean to help Ada rather than complain about the situation, I suggest you go out and write free software in Ada. You may join one of the existing projects, e.g. the Unified Ada Library being discussed in another thread; or you may use the libraries I mentioned for maximum leverage in an application for end users. If you feel that integration is poor between these libraries, then go out and fix that. If you think that they are not portable enough, then port them. Basically, whenever you say "Ada should", replace it with "I will".

And all the while, put a sticker on your software saying "Engineered to perfection with help from Ada", and let the world be in awe of you.

From: Richard Riehle
<adaworks@earthlink.net>
Date: Sat, 17 Apr 2004 00:12:37
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada

Georg Bauhaus wrote:

> What are the requirements in the minds of CS teachers?
> - A university job is not enough gratifying per se, so they need to feel close to the software industry?

For the past several years, I have been teaching in a university where Ada was originally a required subject. When I first began teaching at this institution, my classes were well-attended and students were enjoying the opportunity to learn Ada. The requirement for Ada was eliminated a couple of years ago and attendance in the Ada classes plummeted.

Many of the other faculty members have expressed a distaste for Ada, some suggesting that the sooner it disappears forever, the better. When I recommend that a student use Ada for a thesis project, unless I am the thesis advisor, the student is told that Ada is not appropriate for serious thesis work. "Use Java or C++, but not Ada."

My school is not unique. Throughout academia, Ada is falling victim to a widespread misunderstanding of its value and capabilities. One might think that well-educated faculty members would know better, but that seems not to be the case.

As long as industry is not using Ada, it is difficult to persuade academics to promote it. As long as academia fails to include Ada in the curriculum, it is difficult to persuade industry of its viability. We have something of a stand-off between industy and academia. It does not help that some of our graduate students are bragging about how they are "ripping out all that old Ada code" in this or that system and replacing it with C++.

Some of the largest military contractors have decided to abandon Ada even as they commend its value over competing technologies. "We just cannot hire experienced Ada programmers, and the universities don't teach it anymore." In their view it is cheaper and easier to find C++ programmers. While this is a short-sigthed, perhaps even irresponsible management decision, for weapon systems development, the fact that the Pentagon has abandoned all support for Ada gives those contractors good reason to go a different direction. From their view, the DoD is actually opposed to the use of Ada. It is a wrong point-of-view, but it is widespread.

We need a statement from someone of influence in the U.S. DoD establishment that affirms the value of Ada. Unfortunately, no one will do this. It is not politically expedient. Furthermore, there may not be anyone left in the Pentagon that actually understands the importance of this issue. As a consequence, we are likely to see, over the next decade, a series of software systems, written in C++ that are expensive to create, difficult to maintain, and highly profitable for military contractors. It is an example, in the case of the U.S. DoD, of "grabbing defeat from the jaws of victory." Ada could have been a powerful force for software superiority. Instead, they are allowing contractors to choose whatever technology is expedient, and, I fear, at the expense of long-term software quality.

From: Georg Bauhaus <sb463ba@l1-hrz.uni-duisburg.de>
Date: Sat, 17 Apr 2004 10:29:50 +0000
Subject: Re: No call for Ada
Newsgroups: comp.lang.ada

Maybe things are slightly different in Europe? A few hints make me think that use of Ada in universities is at least a "moving distribution".

From: Warren W. Gay VE3WWG <warren@ve3wwg.tk>
Date: Fri, 16 Apr 2004 15:45:32 -0400
Subject: Re: No call for it
Newsgroups: comp.lang.ada

Ludovic Brenta wrote:

> The problem is that the companies that make up this high-profile market are quite secretive, and do not normally advertise that they use Ada; they just do it with the understanding that Ada is a competitive weapon in their arsenal. ...

Maybe we just need to get someone to fund some prime time TV advertisments along the lines of (perhaps in a Russian accent):

"PSSST! Do you want to know a secret?
It's a pretty good secret..
Did you know that XYZZY Corp. uses Ada?
You can too.
Stay tuned to find out why and how... "

Ravenscar Profile for C/C++?

From: Marc Le Roy <invalide@invalide.com>
Date: Sun, 25 Apr 2004 15:23:32 +0200
Subject: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.c, comp.lang.c++,comp.lang.ada

Ada Ravenscar is a restricted subset of the Ada language that has been defined for real-time software development in safety critical applications. Completed with additional restrictions like the ones defined in the SPARK profile, it allow to build very deterministic applications that support automatic static code analysis and schedulability analysis.

"Guide for the use of the Ada Ravenscar Profile in high integrity systems" http://polaris.dit.upm.es/~str/proyectos/ork/documents/RP_ug.pdf

I would like to know if there is a similar standard for C / C++. I found only MISRA-C and EC++, but they are rather permissive with respect to the Ravenscar Ada profile. Moreover, because the Ada standard covers concepts that are out of the scope of the C/C++ standards, I suppose that an equivalent of the Ravenscar profile in C/C++ should make reference to an RTOS.

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Mon, 26 Apr 2004 07:48:38 +0200
Organization: AdaCL
Subject: Re: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.c++,comp.lang.ada

Ioannis Vranos wrote:

> C++ provides the necessary structures to built very reliable, efficient and mission critical systems. In the above I define what exceptions are expected from each member function, and we can also use the Resoorurce Aquisition is Initializatization technique which the standard library itself also uses.

The problem with safety critical programming in C or C++ is not what is allowed or possible but what should not be allowed and should be impossible. And for that I just need two line:

```
char X[10];
X[10]='A';
```

From: Michiel Salters <Michiel.Salters@logicacmg.com>
Date: 26 Apr 2004 04:06:08 -0700
Subject: Re: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.c++,comp.lang.ada

What's the problem with that code, from a safety perspective? Certainly a C compiler which is supposed to be suited for safety-critical programs will diagnose this. The base C and C++ languages have quite a number of "undefined behavior - no diagnostic required" cases, but a similar profile may very well tighten that to "undefined behavior - must be rejected at compile time".

The base philosophy in C and C++ is that flexibility can be traded for safety, but not vice versa. Certainly, in C++ it is easy to create a verifiable subset. For instance, it is possible to define a range template and with it a <int,0,10> type. The toolset would be hard pressed to prove that the range template is correct and overflow-free. However, this could be proven by humans. The tool chain instead only has to check that all possible overflows are located in this checked range< > code. Together, this would prove that a body of code is overflow-free.

From: Michiel Salters <Michiel.Salters@logicacmg.com>
Date: 5 May 2004 02:40:39 -0700
Subject: Re: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.c++, comp.lang.ada

Vinzent 'Gadget' Hoefler wrote:

> Michiel Salters wrote:

> > Certainly, in C++ it is easy to create a verifiable subset.

> Not quite true. Or maybe the FIASCO project just did not find it yet? URL: http://os.inf.tu-dresden.de/vfiasco/

Quite true. For instance, if I restrict C++ to the subset which allows only

```
int main() { }
```

I know that that subset is perfectly verifiable. For instance,

```
int main() { int i; }
```

is not in the subset. This shows the main problem: Safe and verifiable subsets are more easiliy created by building from the bottom, not stripping from the top. Yet powerful subsets are created by removal of a few features from the full language.

However, it doesn't have a lot to do with the FIASCO project. The goal there was to deal with a lof of known unsafe features. Some were even unsafe by design (e.g. reinterpret_cast<>). This is not an issue when creating safe subsets; reinterpret_cast<> is the first feature to go.

FIASCO is also hindered by a lack of understanding of C++:

[quote] One of the few things that are required is that all built-in integer types are at least 7 bits wide. [/quote]

Now, as range errors in vaiables are a common cause of errors, this is a rather painful lack of understanding.

From: Martin Dowie <martin.dowie@btopenworld.com>
Date: Wed, 5 May 2004 17:44:39 +0000
Subject: Re: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.ada

Vinzent 'Gadget' Hoefler wrote:

> Well, I'd say that SPARK as an verifiable Ada-subset is quite successful in that regard. I doubt that an equally verifiable subset of C++ can ever be implemented. But perhaps that's just me.

No, it's not just you - I have asked a senior Praxis person before if they had plans for a C++ version and they told they had spent plenty time thinking about how one could achieve something worthwhile for C++ and similar to SPARK but they just couldn't think of anything!

Doesn't prove that such a thing could be done, of course, but they are the people in the best position to attempt such a thing!

That was also ~2001 - things may have changed!

From: Peter Amey <peter.amey@praxis-cs.co.uk>
Date: Thu, 06 May 2004 18:22:06 +0100
Subject: Re: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.ada

Your memory serves you well! I don't think things have changed that much. My personal view is that any such subset, if it did exist, would be so restrictive and unnatural to typical C++ users that they would find it unacceptable. Persuading a potential Ada user of the merits of SPARK is a much easier proposition because they have already taken several steps down the early error detection route.

From: Alexander Kopilovitch <aek@vib.usr.pu.ru>
Date: 14 May 2004 19:27:53 -0700
Subject: Re: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.ada

How about SPARK-classes for C++? I mean regular C++ classes, but with attribute SPARK (in GCC you can relatively easily define such additional attributes for classes, it will look something like __SPARK__), which tells that the class conforms with SPARK-imposed restrictions. In other words, SPARK in C++ can be applied for individual classes, which can be mixed in a program (and even in an individual source file) with other (non-SPARK) classes.

Perhaps many C++ programmers will find this approach acceptable, they may perceive it as reasonable and useful compromise. (And there will be nothing heretic in that - even in SPARK applications for Ada it may happen that some packages are for SPARK examination, while others bypass SPARK for some reasons).

From: Martin Krischik <krischik@users.sourceforge.net>
Date: Mon, 26 Apr 2004 07:40:28 +0200
Organization: AdaCL
Subject: Re: "Ravenscar-like" profile for C/C++
Newsgroups: comp.lang.c, comp.lang.c++, comp.lang.ada

Finallylein wrote:

> This discussion, as I already pointed out, belongs in groups like news:comp.programming and news:comp.software-eng. Language subsetting, for whatever purpose, is not defined by the ISO standard for either C or C++, and is not topical here. Nor is safety critical programming.

Finaly a C / C++ programmer who confesses that safety critical programming is in deed off topic in C and C++.

The Nature of Software

From: Mark Lorenzen <mark.lorenzen@ofir.dk>
Date: Sat, 10 Apr 2004 14:32:20 +0200
Subject: Re: "Tracking the Blackout bug"
Newsgroups: comp.lang.ada

Peter Amey writes:

> > Article at "The Register" about the electricity blackout in the Northeast (USA) last year. No directly relevent to c.l.a but interesting since it talks of race conditions etc. which are issues of Ada. http://www.theregister.co.uk/2004/04/08/blackout_bug_report/

> Interesting read. What I do find irritating are quotes such as
"The company did everything it could..."
"We test exhaustively..."
"Unfortunately, that's kind of the nature of software..."
All these statements are untrue and they also reflect a kind of defeatism that I wholly reject (imagine Boeing saying "OK, the wings did fall off, but we tested it a lot and anyway that is just the nature of aeroplanes").
The developers did NOT do everything they could. They could have used the Ravenscar profile in Ada; they could use RavenSPARK; they could have done some model checking of the concurrent parts of the program. They did NOT test exhaustively because it is impossible (/exhaustingly/ I am willing to believe). And software doesn't HAVE to be cr*p!
sigh

It is unbelievable how often I have heard the third statement quoted above. When I try to argue that constructing a software system is just as much an engineering task as constructing a bridge (although much less mature), I am met with disbelief and "hackish" counter-arguments.

These arguments always turn on single point - that programming is an art or maybe even something resembling a magic craft, which can only be learned through years of hacking. And most certainly you can't use any theoretical knowledge when constructing "real" systems (as opposed to the fancy useless university exercises).

Funny enough, the discussion is always about programming and not construction. For some reason, the programming task is regarded as something special and holy - probably the reason why programming language discussions always turn into holy wars.

About Ada Tutorials

From: Georg Bauhaus <sb463ba@l1-hrz.uni-duisburg.de>
Date: Sat, 20 Mar 2004 00:30:07
Subject: Ada Tutorials
Newsgroups: comp.lang.ada

Randy Brukardt wrote:

> Another tool that provides the same functionality, but a different interface, is a lot harder to use -- and for a novice, might be impossible.

By a level of indirection this brings me to Ada tutorials.

The same tool with a different interface is a lot harder to use not because it is more difficult to use, but because

(a) it requires learning

(b) it requires that you forget the old ways.

If the interfaces are similar enough, they will introduce another level of complexity, because you will have to switch between old memories and fresh memories.

Short Ada tutorials are (freely) available for different audiences. I guess some if not all of them are made for programmers with some experience. Is there one that is both short and suitable for studying by people with less experience?

When I had an opportunity to introduce people to computers, or to specific computer programs like text processing programs, it was always refreshing to see that:

(a) people like to learn how things work (e.g. context menues, para styles)

(b) people are capable of using operating system techniques (e.g. folder integration)

But it is somewhat saddening to see that (a) and (b) don't become lasting skills when there are only the equivalent of "left-mouse-button peers", and when they are reluctant to RTFM because it *appears* bulky. Absurd as it may be, when there is a tutorial part in the FM, the thickness of FMs can hide the good short tutorial part. It is a BIG mistake from both a pedagogical and economic point of view not to draw attention to these tutorials because a good tutorial is what gets you going. It prevents frustration, it might lead to adoption. (For example, people get regularly excited when they learn that there is a mechanism in just about any text processing program that allows for boxes of text being placed somewhere on the page without having to play tricks using tables, or worse spaces and tabs... The tutorials introduce this on one or two pages, conceptually, without listing all the steps you need not understand in order to see and try out what a frame is.)

Is there a good short hands-on Ada beginners tutorial that does not talk about Ada's history, safety, the new features, and this and that, but starts in medias res of programming, just makes you want to write programs using the nice-simple-powerful facilities of the language that you have just seen?

So far I have only found Ada presentation slides by R. Dewar (C) 2004 available from the NYU as "course ware", but I guess they lack spoken words and exercises.

The Importance of Bugs

From: Randy Brukardt <randy@rrsoftware.com>
Date: Mon, 12 Apr 2004 17:56:35 -0500
Subject: Re: AdaImgSvr version 0.5 released.
Newsgroups: comp.lang.ada

Patrice Freydiere wrote:

> We are proud to annonce the version 0.5 of AdaImgSvr this new release implements:
> - MySQL database support
> - Win32 service integration
> - Linux deamon utilities
> - Security fixes
> http://adaimgsvr.sourceforge.net
> Enjoy, and feel free for feedbacks

I see that Ada ImgSvr 0.4 made the list of vulnerabilities in this week's at Risk newsletter. (# 04.14b.8)

One problem with the Ada software out there is that it doesn't have enough bugs. :-) Thus it never appears on those weekly vulnerability bug lists, so no one even knows that Ada is being used in many projects. (I learn a lot about software that I never even thought of from those lists.) I don't know of any security bugs found in any of the RRS products in the last number of years -- I half think I need to introduce some, because we need the exposure...

April's Fool Day

From: dan@irvine.com (Dan Eilers)
Newsgroups: comp.lang.ada
Subject: Hacker's Ada FAQ
Date: 1 Apr 2004 10:42:52 -0800

Hacker's Ada FAQ

1 April 2004

Disclaimer:

The editor of this FAQ accepts absolutely no responsibility whatsoever for the "advise" given herein. The solutions have not been rigorously tested, and do not even pretend to comport with accepted software engineering principles. What happens to work on one compiler is almost certain not to work on another compiler, or even on a different release of the same compiler. You have been warned.

1) Ada as a PDL.

Q. I want to use Ada as a PDL (Programming Design Language) but the compiler complains about missing details that I wish to leave TBD for now, such as the initializers for constants. Even worse, this happens in my lowest level packages, preventing me from using the compiler to at least check the syntax of all my packages. Help!

A. AI-00078 confirms that RM 10.1.4(6) allows an implementation to add illegal compilation units to the Ada environment. But since you might forget which constants you have left TBD, it is best to mark them using pragma errors(off). Since this pragma is non-standard, it is best to also use pragma warnings(off).

```
package pak1 is
 pragma Warnings (Off);
private
 pragma Errors (Off);
 max_speed: constant float := …
 upper_bound: constant integer
  := …
 switch: constant boolean := …
 pragma Errors (On);
end pak1;
```

2) Redundant parameter lists

Q. I am used to languages that don't enforce a strict separation of package specifications and bodies. So it seems redundant and potentially error prone to have to repeat the parameters for subprograms. Help!

A. It is acceptable to use pragma errors(off) when a subprogram has more than 10 parameters.

```
package pak2 is
generic procedure p2
 (i,j,k,l,m,n: integer;
  q,r,s,x,y,z: float);
end pak2;
package body pak2 is
 pragma Warnings (Off);
 pragma Errors (Off);
 procedure p2 is separate;
 pragma Errors (On);
end pak2;
```

3) Internal representation of enumerations

Q. I need to get at the internal representation of object of an enumeration type. Pascal offers "union" types, but Ada doesn't seem to allow that. Do I have to resort to unchecked_conversion or non-standard attributes?

A. Try wrapping the union type in a record. Occasionally this will work.

```
function char_to_int
 (ch : character)
 return integer is
 pragma Warnings (Off);
 type union (b: boolean := false)
  is record
 case b is
  when false  => i: integer;
  when true   => c: character;
 end case;
 end record;
 type wrapper is record
  u : union;
 end record;
 type wrapper_ptr is
  access wrapper;
 x : wrapper_ptr := new wrapper;
 pragma Errors (Off);
 j : integer renames x.u.i;
 pragma Errors (On);
begin
 x.u := (b => true, c => ch);
 return j;
end char_to_int;
with text_io; use text_io;
with char_to_int;
procedure main is
begin
 put_line
  (integer'image
   (char_to_int ('A')));
end;
```

[…]

Conference Calendar

This is a list of European and large, worldwide events that may be of interest to the Ada community. Further information on items marked ♦ is available in the Forthcoming Events section of the Journal. Items in larger font denote events with specific Ada focus. Items marked with ☺ denote events with close relation to Ada.
The information in this section is extracted from the on-line *Conference announcements for the international Ada community* at: http://www.cs.kuleuven.ac.be/~dirk/ada-belgium/events/list.html on the Ada-Belgium Web site. These pages contain full announcements, calls for papers, calls for participation, programmes, URLs, etc. and are updated regularly.

2004

☺ June 30 – July 02	16th **Euromicro Conference on Real-Time Systems** (ECRTS'04), Catania, Italy. Topics include: embedded real-time systems; real-time control applications; frameworks and tools for development and analysis; software architectures and languages; design, scheduling, timing and execution-time analysis; validation; etc.
July 05-09	8th **International Conference on Software Reuse** (ICSR-8), Madrid, Spain. Theme: "Software Variability Management for Reusable Software". Topics include: Software generators and domain-specific languages; Quality aspects of reuse, e.g. security and reliability; Success and failure stories of reuse approaches from industrial context; etc.
☺ July 07-09	10th **International Conference on Parallel and Distributed Systems** (ICPADS'2004), Newport Beach, California. Topics include: Parallel and Distributed Systems, Parallel and Distributed Applications and Algorithms, Distributed Operating Systems, Security and Privacy, Dependable Computing and Systems, Real-Time Systems, etc.
July 12-13	**Foundations of Computer Security** (FCS'2004), Turku, Finland. Affiliated with LICS'2004 and ICALP'2004. Topics include: Formal specification, Language-based security, Static analysis, etc.
☺ July 12-15	**OMG** Annual **Workshop on Real-Time and Embedded Distributed Object Computing**, Washington, DC, USA. Topics include: Applying CORBA in any real-time or embedded system; High-confidence, high-availability or safety-critical CORBA applications; Security considerations in real-time or embedded CORBA deployments; Real-Time & Embedded Specifications and Standards; Real-Time & Embedded Product Issues; Real-Time and Embedded Advanced R&D Topics, such as advanced scheduling techniques and high-level real-time programming models; etc.
July 12-16	10th **International Conference on Algebraic Methodology And Software Technology** (AMAST'2004), Stirling, Scotland, UK
☺ July 25-28	23rd Annual **ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing** (PODC'2004), St. John's, Newfoundland, Canada. Topics include: all areas of distributed systems; any aspect of distributed computing, including systems, design, verification, implementation, application, ...; implementation, analysis, evaluation, and deployment of real systems; intersection of security and distributed computing; etc.
August 17-19	11th Nordic **Workshop on Programming and Software Development Tools and Techniques** (NWPER'2004), Turku, Finland. Topics include: tools, methods and languages for programming and software development, etc.
August 19-20	3rd **International ACM-IEEE Symposium on Empirical Software Engineering** (ISESE'2004), Redondo Beach, CA, USA. Topics include: strengths and weaknesses of software engineering technologies; empirical studies of software processes and products; evaluation and comparison of techniques and models (cost estimation, analysis and design methods, testing); reports on benefits derived from using certain technologies; experience management; etc.
August 19-20	9th Australian **Workshop on Safety Related Programmable Systems**, Brisbane, Australia. Theme: "Transport - Can we trust programmable technology?"

August 22-27		18th **IFIP World Computer Congress** (WCC'2004), Toulouse, France. Includes a.o. the following events:
	August 22-27	**IFIP 13.5 Working Conference on Human Error, Safety and Systems Development** (HESSD'2004). Topics include: Aviation, Training, System design maintenance, Design, Methodologies, Formal methods, etc.
	August 26-27	2nd **International Workshop on Formal Aspects in Security & Trust** (FAST'2004). Topics include: Language-based security, Case studies, etc.
	August 27	**Workshop on Architecture Description Languages** (WADL'2004). Topics include: Components, Connectors, Composition; Semantics, Formalization; Verification, Simulation, Test; Tools and Development Environments; Standardization; Industrial Projects.
☺ August 25-27		**Real-Time and Embedded Computing Systems and Applications Conference** (RTCSA'2004), Gothenburg, Sweden. Topics include: quality of service and scheduling; software engineering; fault-tolerance and distributed systems; programming languages and run-time systems; formal methods, and design and analysis tools; case studies; etc.
August 26-28		11th **International Static Analysis Symposium** (SAS'2004), Verona, Italy. Co-located with LOPSTR'04, PEPM'04, and PPDP'04
August 30 – Sep. 04		15th **International Conference on Concurrency Theory** (CONCUR'2004), London, UK. Includes a.o. the following events:
	☺ August 30	**Workshop on Object-Oriented Developments** (WOOD'2004). Topics include: language semantics, type systems, program analysis and verification, design of new calculi and languages, etc.
	September 04	4th **International Workshop on Automated Verification of Critical Systems** (AVoCS'2004)
☺ August 31 – Sep 03		10th **International Conference on Parallel and Distributed Computing** (Euro-Par'2004), Pisa, Italy. Topics include: Support tools and environments; Compilers for high performance; Distributed systems and algorithms; Parallel programming models, methods and languages; etc.
August 31 – Sept 03		30th **EUROMICRO Conference** (EUROMICRO'2004), Rennes, France
September 06-10		12th **IEEE International Requirements Engineering Conference** (RE'2004), Kyoto, Japan. Includes a.o. the following event:
	September 06-07	**International Workshop on Principles of Software Evolution** (IWPSE'2004). Topics include: theory of software evolution; evolution of requirements and environments; architecture for evolution, evolution of architecture; methodology for evolutionary design and development; validation and verification of evolution; environment for evolutionary design; experience reports and lessons learned from evolutionary software systems; etc.
September 08-10		4th **International Conference on Quality Software** (QSIC'2004), Braunschweig, Germany. Topics include: economics of software quality, review, inspection and walkthrough, reliability, safety and security, quality tools, formal methods, static and dynamic analysis, validation and verification, distributed systems, embedded systems, enterprise applications, etc.
September 11-12		1st **International Workshop on Views on Designing Complex Architectures** (VODCA'2004), Bertinoro, Italy. Topics include: Information Security (language-based security, availability, ...); Information Management (component-based software engineering, software reuse, trust management, distributed systems, ...)
September 11-17		20th **IEEE International Conference on Software Maintenance** (ICSM'2004), Chicago, IL, USA.
September 14-16		10th **International Software Metrics Symposium** (Metrics'2004), Chicago, IL, USA. Co-located with ICSM'2004. Topics include: Use and reuse of open source software in industry, Empirical

	studies of open source software development practices, Studies of evolving software systems, Software changeability, Empirical studies evaluating new technologies, etc.
☺ September 15-17	17th **International Conference on Parallel and Distributed Computing Systems** (PDCS'2004), San Francisco, California, USA. Topics include: Reliable Distributed Computing; Languages, Compilers, and Operating Systems; Libraries and Programming Environments; Software Development, Services, Support, Tools; Middleware for Parallel and Distributed Computing; Embedded Systems; Parallel and Distributed Applications; etc.
☺ September 19-22	5th Austrian-Hungarian **Workshop on Distributed and Parallel Systems** (DAPSYS'2004), Budapest, Hungary. Topics include: Distributed and Grid middleware; Parallel and distributed programming languages and methodologies; Software engineering and development tools, environments for parallel systems; etc.
September 20-21	9th **International Workshop on Formal Methods for Industrial Critical Systems** (FMICS'2004), Linz, Austria. Co-located with ASE'2004. Topics include: Verification and validation of complex, distributed, real-time systems and embedded systems; Verification and validation methods that aim at circumventing shortcomings of existing methods in respect to their industrial applicability; Case studies and project reports on formal methods related projects with industrial participation (e.g. safety critical systems, mobile systems, object-based distributed systems); etc.
September 20-22	5th Argentine **Symposium in Software Engineering** (ASSE'2004), Córdoba, Argentina. Topics include: Software Quality; Object Oriented Technology and Theory; Design Patterns; Reuse; Software Understanding; Maintenance and Reverse Engineering; Reliability, Safety and Security; Formal methods; Tools and Development Environments; Education in software engineering; Software Engineering Techniques for Challenging Application Areas such as Distributed Systems, Real-Time Systems, etc.
September 20-24	8th **International IEEE Enterprise Distributed Object Computing Conference** (EDOC'2004), Monterey, California, USA. Topics include: Use and enhancement of middleware platforms; Practical experiences with enterprise distributed object computing; etc.
September 20-25	19th **IEEE International Conference Automated Software Engineering** (ASE'2004), Linz, Austria. Includes a.o. the following event:
	September 20-21 **Software Engineering and Middleware Workshop** (SEM'2004). Topics include: Impact of middleware on software architecture design; Software architecture and architectural styles for middleware-based distributed systems; Selection of middleware; Performance, reliability, security, heterogeneity, scalability issues of middleware; etc.
☺ September 21-24	23rd **International Conference on Computer Safety, Reliability and Security** (Safecomp'2004), Potsdam, Germany. Topics include: Safety Foundations (Fault Tolerance; Distributed and Real-time Systems; Maintenance; Reliability; Formal Methods; Risk Analysis; Open Source Software and Safety; Standards, Guidelines and Certification; Commercial-Off-The-Shelf; Verification, Validation and Testing; ...); Safety Applications (Aerospace and Avionics; Automotive; Medical Systems; Power Plants; Railways; Robotics; Chemical Industry; Process Industry; Programmable Electronic Systems; Accident Reports and Management); Security in Safety-Critical Systems; etc.
☺ September 22-24	GI-Jahrestagung Informatik 2004 - **Workshop "Automotive Software Engineering & Concepts"**, Ulm, Germany
September 22-24	8th **Conference on Formal Techniques in Real-Time and Fault Tolerant Systems** (FTRTFT'2004), Grenoble, France
September 26-30	2nd **IEEE International Conference on Software Engineering and Formal Methods** (SEFM'2004), Beijing, China. Topics include: software specification, validation and verification; software design and refinement; integration of formal and informal methods; fault-tolerant, real-time and hybrid systems; analysis of safety-critical systems; light-weight formal methods; CASE tools and tool integration; application to industrial cases; etc.

September 27-30	24th **IFIP WG 6.1 International Conference on Formal Techniques for Networked and Distributed Systems** (FORTE'2004), Madrid, Spain
September 28-30	28th **IEEE Annual International Computer Software and Applications Conference** (COMPSAC'2004), Hong Kong, China. Theme: "Developing Trustworthy Software Systems" Topics include: Software safety; Trustworthy software; Software fault tolerance; High performance software; Component-based software development; Design patterns; Software certification; Software standards; Software engineering education; Distributed systems; Embedded systems; Enterprise systems; High dependable systems; etc.
☺ October 04-08	18th **International Symposium on DIStributed Computing** (DISC'2004), Amsterdam, the Netherlands. Topics include: distributed programming languages; distributed applications; specification, semantics, and verification of distributed systems; fault-tolerance of distributed systems; cryptographic and security protocols for distributed systems; etc.
♦ October 07-08	**Ada-Deutschland Tagung 2004**, Stuttgart, Germany. Co-located with the Automotive - Safety and Security 2004 Workshop, October 6-7, 2004. Topics include (in German): Methoden und Werkzeuge für zuverlässige Softwaresysteme; Beherrschung der Komplexität in SW-Projekten; UML Profile für zuverlässige Software; Vorgehensmodelle und Lifecycle Management von Systemen; Echtzeitsysteme mit Ada; Sichere Software mit Ada; Ravenscar und weitere Sprachprofile; Erfahrungsberichte über Produktivität, Performance und Kosten in Ada-Projekten; Interoperabilität von Ada und anderen Programmiersprachen; Ada in der Ausbildung; etc.
October 11-15	7th **International Conference on the Unified Modeling Language** (UML'2004), Lisbon, Portugal. Deadline for submissions: July 2, 2004 (tool exhibits), July 19, 2004 (workshop papers), August 30, 2004 (posters and demos)
October 18-20	5th **Conference for Quality in Information and Communications Technology** (QUATIC'2004), Porto, Portugal. Theme: "Quality: a bridge to the future in ICT". Topics include: Economics of quality: costs and savings; Critical success factors in quality improvement; Quality and legal issues; Reliability, safety and security issues; System extensibility: accommodating evolving requirements; Integrating new and legacy systems; Improving the quality of legacy systems; Product reengineering for quality improvement; Forecasting quality characteristics; Quality perception by customers; ICT teaching quality; etc.
October 18-22	**ACM/IFIP/USENIX International Middleware Conference** (Middleware'2004), Toronto, Ontario, Canada. Topics include: Distributed real-time and embedded middleware platforms; Reliable and fault-tolerant middleware platforms; Applications of middleware technologies, including telematics, command and control, avionics, and e-commerce; Novel paradigms, APIs, and languages for distributed systems; Impact of emerging Internet technologies and standards on middleware platforms; etc. Deadline for submissions: July 10, 2004 (posters)
October 18-22	18th Brazilian **Symposium on Software Engineering** (SBES'2004), Brasília, Federal District, Brazil. Topics include: Component-Based Software Engineering; Design Patterns and Frameworks; Software Engineering Industrial Applications; Software Engineering Tools and Environments; Software Maintenance and Reverse engineering; Software Quality; Software Reengineering; Software Reuse; Software Verification, Validation and Testing; etc.
☺ October 24-28	19th Annual **ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications** (OOPSLA'2004), Vancouver, Canada. Topics include: object technology and its offshoots. Deadline for submissions: July 2, 2004 (posters, demonstration proposals, and Doctorial Symposium and Student Volunteers submissions)
October 24-28	3rd **International Conference on Generative Programming and Component Engineering** (GPCE'2004), Vancouver, Canada. Topics include: Generative techniques for Product lines and architectures, Embedded systems, etc.; Component-based software engineering (Reuse, distributed platforms, distributed systems, evolution, analysis and design patterns, development methods, formal methods); Integration of generative and component-based approaches; Industrial applications; etc. Deadline for submissions: July 2, 2004 (demonstrations)

☺ October 25-29	6th **International Symposium on Distributed Objects and Applications** (DOA'2004), Larnaca, Cyprus. Topics include: Enabling Technologies, Middleware, Distributed Objects and Applications, etc.
October 25-29	**International Conference on Practical Software Quality Techniques & Testing Techniques** (PSQT/PSTT'2004 North), Minneapolis, Minnesota, USA. Conference dates changed from August 23-27, 2004. Deadline for submissions: July 9, 2004 (papers, presentations)
October 31 – Nov. 06	ACM SIGSOFT 2004 12th **International Symposium on the Foundations of Software Engineering** (FSE-12), Newport Beach, California, USA. Topics include: Component-Based Software Engineering; Empirical Studies of Software Tools and Methods; Generic Programming and Software Reuse; Software Engineering and Security; Software Engineering Tools and Environments; Software Metrics; Software Reliability Engineering; Software Safety; Specification and Verification; etc.
November 02-05	3rd **International Symposium on Formal Methods for Components and Objects** (FMCO'2004), Leiden, the Netherlands
November 02-05	15th **IEEE International Symposium on Software Reliability Engineering** (ISSRE'2004), Saint-Malo, Bretagne, France. Theme: "Achieving Software Dependability through Model-Driven Engineering". Deadline for submissions: July 1, 2004 (industry practice), July 10, 2004 (student papers, fast abstracts)
November 04-06	2nd Asian **Symposium on Programming Languages and Systems** (APLAS'2004), Taipei, Taiwan. Topics include: semantics and theoretical foundations; type systems and language design; language interpreters and compilers; program analysis, optimization and transformation; models and tools for parallel and distributed systems; language support for security and safety; domain-specific languages and systems; storage management techniques; software development methods and systems; techniques for embedded and mobile code; process algebra and concurrency; etc.
November 09-12	11th **Working Conference on Reverse Engineering** (WCRE'2004), Delft, the Netherlands. Topics include: Program analysis and slicing; Program transformation and refactoring; Legacy systems; Transitioning to software product lines; Documentation generation; Pre-processing, parsing and fact extraction; Reengineering patterns; Traceability recovery; Reverse engineering economics; UML and roundtrip engineering; Program comprehension; Reverse engineering tool support; etc. Deadline for submissions: July 7, 2004 (workshops), October 13, 2004 (tool descriptions)
November 10-12	**European Software Process Improvement Conference** (EuroSPI'2004), Trondheim, Norway
♦ November 14-18	2004 **ACM SIGAda Annual International Conference** (SIGAda'2004), Atlanta, Georgia, USA. Topics include: safety and high integrity issues, real-time and embedded applications, Ada & software engineering education, Ada in other environments such as XML and .NET, Ada and other languages, metrics, standards, analysis, testing, validation, and quality assurance, etc.
November 18-19	**CoLogNet** / **Formal Methods Europe Symposium on Teaching Formal Methods** 2004, Gent, Belgium. Topics include: experiences of teaching Formal Methods, both successful and unsuccessful; educational resources including the use of books, case studies and the internet; the integration, or otherwise, of FMs into the curriculum; the advantages of FM trained graduates in the workplace; changing attitudes towards FMs in students, academic staff and practitioners; etc.
November 22-26	John Robinson & Associates - **Public Ada Programming Course**, Cheltenham, UK. A practical course with two streams covering Ada 83 and Ada 95.
November 30 – Dec. 12	11th Asia-Pacific Software Engineering Conference (APSEC'2004), Busan, Korea. Topics include: Software Design Methods, Software Testing, Verification and Validation, Program Analysis, Software Maintenance, Software Development Methodology, Metrics and Measurement, Software Quality Assurance, Object-Oriented Technology and Design Patterns, Components Based Software Engineering, Product Line Engineering, Distributed and Parallel Software Engineering, Embedded & Real-Time Software Systems, Standards and Legal Issues, Software Engineering Education, etc.

☺ December 05-08	25th **IEEE Real-Time Systems Symposium** (RTSS'2004), Lisbon, Portugal. Topics include: QoS support; Real-time systems middleware; Security and survivability; Real-time and dependability; Compiler support; Embedded operating systems; Software engineering; RT programming languages; Scheduling; Formal methods; Case-studies; etc.
December 10	Birthday of Lady Ada Lovelace, born in 1815. Happy Programmers' Day!
☺ December 13-15	2nd **International Symposium on Parallel and Distributed Processing and Applications** (ISPA'2004), Hong Kong, China. Topics include: Parallel/distributed system architectures; Parallel/distributed algorithms; Reliability, fault-tolerance, and security; Performance evaluation and measurements; Tools and environments for software development; Distributed systems and applications; High-performance scientific and engineering computing; etc. Deadline for submissions: July 1, 2004 (papers)
☺ December 15-17	8th **International Conference on Principles of Distributed Systems** (OPODIS'2004), Grenoble, France. Topics: theory, specifications, design and implementation of distributed systems, including: real-time and embedded systems; distributed and multiprocessor algorithms; communication and synchronization protocols; self-stabilization, reliability and fault-tolerance; specification verification of distributed systems; security issues in distributed computing and systems; etc. Deadline for submissions: July 31, 2004

2005

January 03-06	*Software Technology Track* of the 38th Hawaii **International Conference on System Sciences** (HICSS-38), Big Island of Hawaii, USA. Includes mini-tracks on: Strategic Software Engineering; Adaptive and Evolvable Software Systems: Techniques, Tools, and Applications; etc.
February 07-11	4th **International Conference on COTS-Based Software Systems** (ICCBSS'2005), Bilbao, Spain. Deadline for submissions: July 16, 2004
April 02-10	**European Joint Conferences on Theory and Practice of Software** (ETAPS'2005), Edinburgh, Scotland, United Kingdom. Event includes: conferences from 4-8 April, 2005, satellite events on 2-3 and 9-10 April, 2005
☺ April 05-08	2nd **Jahrestagung Fachbereich "Sicherheit - Schutz und Zuverlässigkeit"**, Regensburg, Germany. Event includes: special session "Informationssicherheit im Automobil", workshop "Privacy Respecting Incident Management", etc. Topics include (in German): Vertrauenswürdige Softwarekomponenten; Zuverlässigkeit und Fehlertoleranz in Hardware- und Softwaresystemen; Formale Techniken, Modellierung, Spezifikation und Verifikation; Betriebssicherheit unter extremen Bedingungen; Standards und Normung; Sicherheitsevaluation und -zertifizierung; Kosten von Sicherheit; etc. Deadline for submissions: October 15, 2004
☺ May 15-21	27th **International Conference on Software Engineering** (ICSE'2005), St Louis, Missouri, USA. Topics include: Software architectures and design; Software components and reuse; Software security; Software safety and reliability; Reverse engineering and software maintenance; Software economics; Empirical software engineering and metrics; Distribution and parallelism; Software tools and development environments; Programming languages; Object-oriented techniques; Embedded and real-time software; etc. Deadline for submissions: September 1, 2004 (papers), October 47, 2004 (tutorial and workshop proposals), December 6, 2004 (doctoral symposium, research demonstrations)
♦ June 20-24	10th **International Conference on Reliable Software Technologies** - Ada-Europe'2005, York, UK. Sponsored by Ada-Europe, in cooperation with ACM SIGAda (approval pending).
July 18-22	13th **International Symposium of Formal Methods Europe** (FM'2005), Newcastle upon Tyne, UK. Topics include: introducing formal methods in industrial practice (technical, organizational, social, psychological aspects); reports on practical use and case studies (reporting positive or negative experiences); tool support and software engineering; environments for formal methods; etc.

Standardizing the Next Version of Ada

James W Moore

Convener, ISO/IEC JTC 1/SC 22/WG 9

The MITRE Corporation, 7515 Colshire Drive, H505, McLean, VA 22102, USA; Tel:+1.703.883.7396; email: James.W.Moore@ieee.org (The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the author.)

Abstract

Ada is a mature language but not a perfect language. As time passes, we discover portions of the language specification that need clarification or, occasionally, contain errors. We also discover better ways in which the language can support programming in both old and new application areas. A major revision of the original Ada language standard was completed in 1995. A corrigendum to the standard was published in 2001. Currently, work is underway to produce an amendment to the standard. That amendment will probably be completed during 2005. This article provides an overview of the international standards process and how the amendment to the Ada language standard will be accomplished.

Background

When discussing standards, it is important to realize that (in English, at least) the word "standard" can have multiple meanings. The word can apply to a range of concepts from proprietary products that have achieved widespread market share, e.g. the Microsoft Windows "standard", to specifications developed through a consensus process intended for widespread implementation in the products of many companies, e.g. the Ada language standard. Furthermore, one should realize that even consensus standards are produced in a variety of ways by a variety of organizations, ranging from professional societies (like the Institute of Electrical and Electronics Engineers) to corporate consortia (like the Object Management Group) to organizations established by treaties among nations (like the International Telecommunications Union or NATO).

A suitable classification for this article is the distinction between *de facto* and *de jure* standards. De facto standards are simply specification or products that have achieved widespread acceptance in the marketplace. De jure standards are specifications produced by organizations that are legally chartered to create standards. Most countries have a single governmental organization that produces standards. The international standards organizations—the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC)—were created by an agreement among these "national bodies".

The situation in the US is more complicated because the US doesn't have a governmentally designated standards organization. Even the American National Standards Institute (ANSI), which serves as the US national body, is only an umbrella for several hundred organizations who agree to follow ANSI's rules in producing their own standards, which are then endorsed by ANSI. Fundamentally, though, any organization in the US can create a "standard" if it follows a few rules regarding openness, due process, and consensus to prevent running afoul of laws regarding antitrust. In the US, we can regard standards produced by members of ANSI as de jure standards, while those produced by others as de facto standards.

International standards development

As I mentioned before, there are two international standards organizations. The International Electrotechnical Commission (IEC) produces standards for electrical and electronic equipment. The International Organization for Standardization (ISO) produces standards for everything else. Each subdivides its scope hierarchically via Technical Committees and Subcommittees. When the first Ada specification was completed, the hierarchical committee structures of ISO and IEC were completely disjoint.

Circa 1986, ISO discovered that IEC was creating standards for *computers* and IEC discovered that ISO was creating standards for *computing*. To avoid conflict, they created their first and only Joint Technical Committee (JTC1) and gave it the scope of "information technology". JTC1 created a number of subcommittees, including SC22 which is responsible for programming languages. Standardization of a number of existing languages was assigned to various Working Groups of SC22. A new working group, WG9, was created to process the international standardization of the ANSI Ada standard.

Despite the tidy hierarchical subdivision of scope, it is clear that information technology standards must relate to other standards committees outside the scope of JTC1. For example, the famous ISO 9000 standards for quality management are produced in an ISO Technical Committee, TC176. Treatment of dependability and safety is performed by two IEC committees, TC56 and SC65A, respectively.

At all levels of ISO and IEC, the participants are "national bodies" rather than individuals. The individuals who attend meetings are required to represent national positions that are developed by a variety of mechanisms in the different nations.

ISO and IEC once had distinct procedures—"directives"—for processing standards work. JTC1 wrote a set of directives that were intended to be a compromise between the two. Although the directives differ in detail, they all describe a similar process. In short, standards are drafted in working groups, receive technical approval by a subcommittee, and receive management approval by a technical committee. For Ada standardization, the jargon is that:

- A Working Draft (WD) is developed by WG9. When consensus is reached, the document is …
- Registered as a Committee Draft (CD) at the SC22 level. Committee Drafts undergo successive balloting periods that are 3 months in duration. Technical comments are accepted from national bodies and the document is modified until consensus is reached. The Final Committee Draft (FCD) ballot is an extra month in length allowing national bodies to perform more extensive inquiries regarding the acceptability of the standard. When the FCD is approved by 75% of the national bodies, the document is forwarded for …
- Balloting as a Final Draft International Standard (FDIS) at the JTC1 level. This ballot is only two months in duration and provides management approval. No technical comments are accepted. An approved document is …
- Published as an International Standard (IS).

Working Groups can also develop documents that are not standards and are called Technical Reports. Their approval follows a process that is similar but has different names for the stages.

Although the process appears complicated, it is actually possible to progress a standard rapidly—once consensus has been reached. The "speed of light" for the voting process is about one year.[1] To progress rapidly, a working group must ensure that technical consensus is reached within the working group before the document is progressed to the subcommittee level. Of course, it also helps if the delegates to the working group are careful to advise their national bodies of the existence of consensus. If this is done carefully, then the voting above the working group level can be considered as merely an administrative exercise to confirm the existence of consensus.

[1] This rapid speed is not merely theoretical. The Ada committee, ISO/IEC JTC1/SC22/WG9, actually processed ISO/IEC 18009, Conformity Assessment of Ada Language Processors, in 13 months—after reaching consensus on its contents.

According to the Directives, a JTC1 standard must be reconsidered every five years for confirmation, revision, withdrawal or stabilization (retention without maintenance). Besides revision, though, there are two other ways to update a standard. A Corrigendum (COR) permits the correction of defects in the wording of a standard and is approved by an abbreviated process that delegates final authority to the Subcommittee. The Amendment process allows more extensive technical changes to be made without reopening the entire standard for revision. Amendments are approved by a process similar to the approval of a standard. Following a Working Draft, the stages of approval are called Preliminary Draft Amendment (PDAM), Final Preliminary Draft Amendment (FPDAM), and Final Draft Amendment (FDAM), resulting in an Amendment (AMD). Corrigenda or Amendments may be published as distinct documents or may be combined with the base standard.

Ada standardization

The original Ada standard was published in 1983. It was created through a consensus process facilitated by contracts from the US Department of Defense (DoD) and approved as an ANSI standard via the "canvass" process that essentially allows anyone who claims an interest to vote yes or no. In 1987, after the creation of JTC1, the specification and its French translation were adopted as ISO/IEC 8652 and assigned to a new SC22 working group, WG9. A revision of the language commenced shortly thereafter, again facilitated by contracts from the US DOD, resulting in publication of the revised standard in 1995.

From the beginning, the Ada language has always had a process for dealing with perceived or actual flaws in the specification of the language. Even the initial version of the specification provided directions for submitting Ada comments. These comments were handled by a designated group of reviewers. Selected comments resulted in the drafting of Ada Issues. Some Ada Issues were politely rejected, some resulted in explanations, some resulted in clarifications, and some resulted in implicit changes to the specification. The implicit changes to the specification were handled via notification to compiler suppliers and modifications to the compiler "validation" suite of test cases intended to check conformance to the standard.

In the years following the ISO adoption of the Ada standard, ISO developed procedures for dealing with perceived "defects" in standards. National bodies were permitted to write "defect reports" which were given to a designated "editorial team". If the editorial team determined that the standard was, in fact, flawed, then a Corrigendum to the standard would be written.

After the completion of the 1995 standard, and the subsequent end of US DOD subsidies, the ISO working group for Ada became the focal point for continued improvement of the language. WG9 wrote a standard regarding compiler conformance assessment—ISO/IEC 18009. The traditional Ada defect correction process was formalized to comply with the rules of the JTC1 defect correction process.[2]

The Ada standards committee took advantage of the new formal correction process to produce a corrigendum to the Ada standard in 2001. The mechanism was suitable for the addition of clarifying text and the correction of wording errors in the standard. However, the mechanism was not suitable for making substantive changes to the language to address changing user requirements.

Six years of experience since the 1995 revision, though, had indicated the need for a technical refresh. Revision of the language standard was considered but rejected for concern that it might signal dissatisfaction with the language. Instead, WG9 decided to prepare an amendment to the language standard. WG9 customarily delegates document preparation to subgroups (called "Rapporteur Groups" in the jargon). The responsibility to draft the amendment was delegated to the Ada Rapporteur Group (ARG), the same group that had developed the Corrigendum. The ARG is currently chaired by Pascal Leroy and the amendment editor is Randy Brukardt. The ARG's members are language experts nominated by the national bodies and by the two professional societies that are liaisons to WG9—SIGAda and Ada-Europe.

From the beginning, it was recognized that the Ada user community should not be destabilized by a concern that the language definition would be subject to widespread change. It was decided that the extent of language change should be strictly bounded by providing a set of "instructions" to the ARG regarding the preparation of the amendment.

The instructions to the ARG

In October 2002, WG9 agreed on the wording of a set of instructions (WG9 N412) [1] to the ARG regarding their preparation of a draft amendment to the Ada language standard. Despite spirited discussion, broad agreement was achieved and the instructions were approved by all six of the national bodies which cast a ballot. This section discusses those instructions and offers my interpretation of the significance of those instructions.[3]

WG9 specified that the

> *purpose of the Amendment is to address identified problems in Ada that are interfering with Ada's usage or adoption, especially in its major applications areas (such as high-reliability, long-lived real-time and/or embedded applications and very large complex systems). The resulting language changes may range from relatively minor, to more substantial.*

In restricting the amendment to "identified problems", WG9 rejected an overall refresh of the language design or the exploration of speculative changes to the language. The phrase "usage or adoption" is intended to suggest appeal to both current and new users within identified application areas—"high-reliability, long-lived, real-time, and/or embedded applications and very large complex systems". The final sentence steers a middle course between sanctioning extensive change and prohibiting it.

WG9 agreed that two changes should be made to the language:

- *inclusion of the Ravenscar profile,*
- *inclusion of a solution to the problem of mutually dependent types across packages.*

Beyond those two, WG9 chose to state its desires more generically.

WG9 designated two categories of improvement:

> *(A) Improvements that will maintain or improve Ada's advantages, especially in those user domains where safety and criticality are prime concerns;*
>
> *(B) Improvements that will remedy shortcomings in Ada.*

In item (B), the phrase "with respect to other languages" was considered, but rejected, suggesting that the ARG should not offer a feature-by-feature competition with other languages. Of course, item (A) calls particular attention to Ada's existing advantages for applications involving safety and other critical properties, and suggests building on the language's strengths in those areas.

The next part of the instructions elaborates further on these categories:

> *Improvements in the real-time features are an example of (A) and should be considered a high priority. Improvements in the high-integrity features are an example of (A) and should be considered a high priority. Features that increase static error detection are an example of (A) and should be considered a priority, but less important than the two listed above. Improvements in the facilities for interfacing to other languages are an example of (A) and should be considered. Improvements in the object-oriented features—specifically, adding a Java-like interfaces feature and improved interfacing to other OO languages—are an example of (B) and should be considered.*

I interpret this language as providing three levels of priority:

- "High Priority"
  - Real-time features
  - High-integrity features
- "A Priority, but less Important"
  - Increased static error detection

[2] This was easy because the ISO process was derived in part from the Ada process.

[3] Parts of this section are based on [2].

- "Should be Considered"
    - Interfacing to other languages
    - Object-oriented features—specifically, adding a Java-like interfaces feature and providing improved methods for interfacing to other OO languages

The list is notable, not only for the prioritization, but also for what is missing. WG9 considered but rejected adding "design by contract features" to the list. No other categories of features were offered during the discussion of the instructions.

The prioritization advises the ARG of the nature of desired improvements, but does not restrict them in the choice of technical solutions for improvement. It is assumed that the specific proposals come from the "Ada Issues", including those submitted from outside WG9 and those written by the ARG itself.

In looking for good solutions, a period of time was provided for outside submissions. The instructions included an appropriate schedule, beginning with the approval of the instructions in October 2002 and continuing with:

- *December 2002: Presentation at SIGAda, providing for discussion groups and feedback.*
- *June 2003: Similar presentation at Ada-Europe*
- *September 2003: Receipt of the final AIs from groups other than WG9 or delegated bodies*
- *September 2003: Presentation at the International Ada Real-time Workshop (IRTAW)*
- *Autumn 2003: Presentation at SIGAda*
- *December 2003: Receipt of the final AIs from WG9 or delegated bodies*

Nearly a year was provided for outside submissions. Three additional months were provided for groups specifically delegated to write proposals or improve existing proposals, for example, the IRTAW.

Of course, it is the task of the ARG to select among the proposals for language improvement. In the instructions, WG9 provided some criteria for making that selection:

- *Implementability (vendors' concerns). Can the proposed feature be implemented at reasonable cost?*
- *Need (users' concerns). Does the proposed feature fulfil an actual user need?*
- *Language stability (users' concerns). Would the proposed feature appear disturbing to current users?*
- *Competition and popularity. Does the proposed feature help improve the perception of Ada, and make it more competitive with other languages?*
- *Interoperability. Does the proposed feature ease problems of interfacing with other languages and systems?*
- *Language consistency: Is the provision of the feature syntactically and semantically consistent with the language's current structure and design philosophy?*

As before, items missing from the list can be as informative as items present in the list. WG9 considered adding the criterion of "uniqueness and innovation" to the list, but rejected it—another indication that WG9 was interested in addressing identified problems rather than expanding the language into new areas.

Of course, it is now well known that additions or changes to languages cannot be made without some impact on users. For example, words that were once legitimate identifiers might be pre-empted as keywords for new language features. The Ada 95 revision was extremely conservative in dealing with backward incompatibility.[4] WG9 wanted to provide a little more freedom to the ARG in preparing this amendment, but it is difficult to write that without providing license. The following wording was used in the instructions:

> *In order to produce a technically superior result, it is permitted to compromise backwards compatibility when the impact on users is judged to be acceptable.*

It should be noted that this wording was approved by a very narrow margin in WG9, suggesting that proposed incompatibilities may be judged skeptically.

In the 1995 language revisions, some subjects were relegated to "secondary standards" rather than being incorporated into the annexes of the Ada language standard.[5] Since then, a consensus has emerged that placing a subject in a secondary standard, in effect, dooms it to obscurity. WG9 decided that the mistake should not be repeated and offered instructions on that subject:

> *The use of secondary standards should be minimized; secondary standards should be proposed only when they would include material so important as to require standardization but so voluminous as to preclude inclusion in the Ada language standard. In particular, material similar to the current ISO/IEC 13813, Generic Packages of Real and Complex Vector and Matrix Type Declarations and Basic Operations for Ada, should be incorporated into the language standard.*

[4] For the 1995 revision, WG9 attempted to catalogue every possible incompatibility, assess its impact on users, and reduce the impact to insignificance.

[5] The term “secondary standard” is not used by the JTC1 directives. I think it is peculiar to the Ada community.

The ARG instructions also provided a schedule for the approval of the amendment:

- *June 2004: WG9 approval of the scope of amendment (perhaps by approving AIs, perhaps by reviewing draft amendment)*
- *Informal circulation of draft, receipt of comments and preparation of final text*
- *Spring 2005: Completion of proposed text of amendment to be contributed to WG9*
- *Mid 2005: WG9 email ballot*
- *Third Quarter 2005: SC22 FPDAM ballot*
- *Late 2005: JTC1 FDAM ballot*

As of the writing of this article (May 2004), the preparation of the amendment is on schedule. The ARG Chair, Pascal Leroy, has prepared a document [3] describing the scope of the amendment and has tabled it for the June 2004 meeting of WG9. The document uses AIs to define the scope of the amendment. This is not intended to suggest that the AIs are yet approved, but only to indicate that a solution to the problem described by the AI is within the scope of the amendment. The AIs are grouped and aligned with the instructions, providing assurance that the ARG is carrying out WG9's intent.

It is important to remember that the "scope" of a standard or an amendment is a boundary. The document should not venture outside the scope but is not required to explore all of the available space within the scope. So the listing of an AI in this scope document does not presume that the AI as currently written will be included in the amendment. Furthermore, it doesn't even guarantee that the problem described in the AI will be addressed in the amendment. All of that is determined later in the process.

We can anticipate that WG9 will approve this scope document—perhaps with some changes—during its June meeting 2004, or shortly thereafter via an email ballot. During the succeeding months, the ARG will continue to work on the AIs and begin drafting the text of the amendment. The next milestone is spring of 2005 when the ARG is requested to submit the draft text of the amendment. It will be circulated to WG9 for comment and consideration by the national bodies. Approval of text would be done by an email ballot during mid-2005.

At that point, the process becomes more formal and proceeds to voting by SC22 and then JTC1. We hope to process the standard through balloting by the parent bodies at the ISO "speed of light" that was described previously. So it is important that all technical issues are resolved within the ARG and confirmed by WG9 prior to progressing the amendment to the subcommittee level. In the past, WG9 has been very successful in anticipating the concerns of SC22 and JTC1 and communicating our technical judgment to them via the national bodies represented in WG9. It is reasonable to hope that the amendment can be approved without further changes and published early in 2006.

Summary

An amendment to the Ada language standard is being prepared for publication early in 2006. It will address many of the issues of usage that have arisen since the language's revision in 1995 but which could not be addressed by the 2001 corrigendum. The emphasis of the amendment is to address identified problems rather than provide a broad update of the language. Perhaps the most notable result of the amendment will be repeated emphasis on safety and criticality as Ada's most important areas of application.

References

[1] ISO/IEC JTC 1/SC 22/WG 9, *Instructions to the ARG for Preparation of the Amendment to ISO/IEC 8652,* WG9 document N412, 10 October 2002. http://www.open-std.org/jtc1/sc22/wg9/n412.pdf. Republished in Ada User, 25:1, March 2004, page 27.

[2] James W. Moore, *Convener's Comments on Instructions to the Ada Rapporteur Group from SC22/WG9 for Preparation of the Amendment to ISO/IEC 8652,* December 2002 SIGAda conference, WG9 document N423, 20 December 2002. http://www.open-std.org/jtc1/sc22/wg9/n423.pdf. Republished in Ada User, 25:1, March 2004, pages 28-29.

[3] Pascal Leroy, *Proposal for Defining Scope of Amendment to ISO/IEC 8652:1995,* WG9 document N437, March 2004. http://www.open-std.org/jtc1/sc22/wg9/n437.pdf. Republished in Ada User, 15:1, March 2004, pages 30-31.

Ada Academic Initiative in Paris

Louise Arkwright

Academic Program Manager, ACT Europe, 8 rue de Milan, 75009 Paris. arkwright@act-europe.fr

On 13th February 2004, a group of Ada Academic experts met at the AdaCore European Headquarters in Paris to review the current uses of Ada for educational purposes in European universities, and to explore ways in which academics and AdaCore can team up to promote and extend the use of Ada.

The main objectives of this initiative are to:

- Encourage and extend the use of Ada in Academia by providing a quality assured software packages, among other material, to facilitate Ada programming for students.
- Create a collaborative platform for the Ada academic community enabling it to access support across several fields (technology, advocacy, teaching materials, etc) as well as facilitating contribution of ideas.
- Generate stronger links between academia and the professional Ada community.

There is a clear need for grouped resources for academic support materials via the provision of website hosting for material and Ada libraries. Such a shared site will focus the energies of the Academic community and lead to the development of software for pedagogical and industrial uses.

The Ada Academic Working Group

To this end, and as a result of the workshop, several proposals are currently being acted upon. Notably, AdaCore is preparing an academic offer to include an integrated binary distribution for Windows, Solaris and Gnu/Linux, as well as source releases for various packages. Also being compiled by the members of the Working Group are materials such as source packages for additional tools, teaching materials (slides, articles, books), laboratory case studies and technical papers. A call for educational material, which will be made freely available as part of the GNAT package to be accessed via the AdaCore website, is currently underway. It is planned to launch the special integrated academic package for the start of the 2004/2005 academic year.

The team at work

Much of this material has already been prepared and AdaCore will be holding a follow-up session at Ada Europe 2004 on 16th June for review. All interested academics are welcome to join this meeting which will continue in the path of a worthwhile collaboration set to grow to include more teachers in the future.

Practical Experiences of Safety- and Security-Critical Technologies

Peter Amey and Adrian J Hilton

Praxis Critical Systems Ltd., 20 Manvers Street, Bath BA1 1PX; Tel: +44 1225 823761; email: {peter.amey/adrian.hilton}@praxis-cs.co.uk

Abstract

In this article we identify the special properties of systems intended for use in ultra-reliable domains and the qualitative shift in development methods needed to achieve those properties. The advantages (and disadvantages) of Ada are introduced in the contexts of the ISO HRG report on High-Integrity Ada and of the SPARK sub-language. The demands of common, important development standards are described together with appropriate and cost-effective techniques for meeting them. Finally project experience illustrating successes in meeting the main standards is discussed.

Keywords: high integrity, safety, security, case study, SPARK, Ada.

1 Introduction

Safety- and security-critical systems have a fundamental common property, that they are not just correct and reliable but that they can be *shown* to be correct and reliable. Most modern critical systems contains software, and often (though not always) the system relies on the operation of the software to achieve safety or security. In this case the software is designated *safety-critical* or *security critical*, and is often termed generically *high-integrity* software.

High-integrity software is code where reliability is more important than efficiency, cost, time-to-market or functionality. Examples are security systems, financial systems where down-time incurs a large cost, and cases where a costly one-off mission capability may be lost such as a Mars Lander.

The unique characteristic of high-integrity software is the need to be able to show, before there is any service experience, that a system will meet its requirements. It is a problem qualitatively different from normal software, since the standards involved may be very high (e.g. requiring the equivalent of no more than 1 failure every 114,000 years of operation). It is not enough to be just "more careful" in development!

In this article we consider how such assurance may be obtained for the software component of a system, in particular for software components written in Ada. We draw on our experiences with illustrative case studies in developing and verifying safety-critical and security-critical systems.

2 Methods of Verification

We define the following terms:

- *verification* as "the process of determining that a system (or its component) meets its specification";
- *validation* as "the process of determining that a system is appropriate for its purpose"; and
- *certification* as "persuading an external regulatory body that a set of specific requirements have been met and/or processes followed".

Techniques of verification include:

- inspections / reviews, e.g. requirements or code;
- testing, e.g. requirements tests or unit tests; and
- analysis e.g. flow analysis, model-checking or partial program proof.

Inspections are uniquely flexible, since the primary tool is the engineer, and can be done early on. However they are inherently informal and fallible, and what inspection is feasible is limited by the *semantic precision* of the object being inspected and the *semantic gap* between the objects being compared.

(Dynamic) testing spans the entire development testing and can potentially identify errors in requirements, specifications, code, the compiler and the hardware. However, exhaustive testing is rarely feasible or even possible, and for high-integrity systems the high levels of assurance required the testing time becomes impractical. Littlewood [4] and Butler [5] note the limitations of Bayesian testing and the implications for reliability of any failures during a very long testing process.

There are also the practical problems arising from the need for a complete or mostly complete system before testing can begin. The testing phase of a system development is frequently a bottleneck for the entire development, and over-running testing is an expensive risk for most projects.

Analysis, by contrast, can be conducted early on in the development process and can establish properties that cannot be shown statistically, such as the proof of absence of run-time errors or freedom from timing deadlocks. However it can only *compare* objects (e.g. the code against the specification), and what can be achieved is limited by the precision of the descriptions and notations used.

Showing "correctness" is therefore harder than building correct systems; the key is to use a range of verification techniques. Since bug detection and correction is expensive, the focus should be on:

- preventing bugs from occurring;
- using techniques to find bugs early on; and
- using final testing as a *demonstration of correct behaviour* rather than as a method of finding bugs.

3 Reliable programming

The language in which we choose to program will affect the way we think about the program. A developer is less likely to think about abstractions if programming in machine code or assembler; she will think more in terms of the target machine if programming in C. For larger systems, language support for abstraction and encapsulation is vital if the program is to remain manageable and verifiable.

3.1 Formality and uncertainty

Machine code is *formal*, in that the target machine provides an operational semantics for it. However we cannot normally reason about the code, we can only observe its behaviour. Early reasoning about program correctness saves money and reduces risk, but is hampered by the lack of formality of most programming languages.

Typical causes of uncertainty in programming languages include deficiencies in the language definition (e.g., the semantics of integer division in C) and deliberate implementation freedoms (e.g. order of evaluation of expression components and parameter passing mechanisms). This leads to either:

- *ambiguity*, where program behaviour cannot be predicted from the source code; or
- *insecurity*, where violations of a language rule cannot be detected.

Ambiguities are resolved by compiler authors. Insecurities are left for the user to discover. Neither of these situations are attractive, but there are possible solutions:

- invent new languages without these problems;
- work with *dialects* associated with compilers; or
- use logically coherent language *subsets* to overcome ambiguities and insecurities.

When inventing new languages the very small user base leads to poor or non-existent tools, staff shortage and a lack of training support; this approach is impractical for most projects.

3.2 Dialects

When using a dialect, the first step is to find out what your compiler does. Its behaviour can then be documented and included in coding standards and review checklists. This can be an effective solution, but there are problems.

First, the compiler's behaviour must be known; it must also be consistent. If it changes with new releases, or between the host and the target, then the dialect may be invalid. Developers must also know when it matters; we must recognise that implementation-dependent behaviour is present and know what the compiler's behaviour is in this case.

You cannot expect to avoid all anomalous behaviour by this method, but it may be valuable in special cases e.g. small, specialised processors where only one company provides support, or for an established compiler where a small number of problems are known but service experience provides confidence in the rest of its behaviour.

3.3 Subsets

The need for subsetting was described by Wichmann [6]: "No currently standardised language could be recommended without reservation for the most critical applications without subsetting". To be useful, however, language subsets need to be:

- simple;
- application oriented;
- predictable;
- formally verifiable; and
- be supported by sound tools.

A subset comprises a base language, a set of troublesome language features which are removed, a set of limitations on the way that remaining features may be used, and optionally the introduction of *annotations* to provide extra information. We can construct subsets that vary on four axes: precision (security and lack of ambiguity), expressive power, depth of analysis possible and the efficiency of the analysis process. There are complex trade-offs in this model.

The fundamental trade-off is between the discipline which programmers accept in order to reduce bug *insertion*, and the effort which they are prepared to make in bug *detection*. For example, unrestricted C provides little or no protection from bug insertion, whereas Ada requires extra discipline (e.g. strong typing) which reduces the bug insertion rate.

A qualitative shift in what is possible will only occur when the precision of the subset becomes exact.

3.4 MISRA C

MISRA-C is a C subset defined by the UK motor industry research association (MIRA) and its associated software reliability association (MISRA). It comprises 127 "rules" presented in the form of a coding standard, and its designers regard it as suitable for "SIL 3" systems; they recommend Ada or Modula-2 if available.

The base language of MISRA C is ISO/IEC 9899:1990 + the 1995 technical corrigendum. It removes troublesome C features such as pointer arithmetic, and imposes limitations such as all switch statements having to have a final default. It does not use extra annotations.

There was no attempt to make the MISRA C subset logically coherent and free from ambiguity and insecurity, and its machine checkability is well short of 100%. Some rules cannot be machine-checked at all, e.g. Rule 4: "Provision should be made for appropriate run-time checking" and the exact meanings of some other rules are unclear and much debated. Nevertheless, following the subset rules should prevent many common C errors.

3.5 The Ada HRG

The Ada HRG is an ISO committee under WG9 to investigate high-integrity Ada issues, responsible for interpreting and developing Annex H of the LRM and focused on developing Ada in the high-integrity field. Its members include tool vendors, users, Government (e.g. the UK Ministry of Defence) and academics.

HRG report ISO/IEC JTC1/SC22/WG9 "Programming Languages – Guide for the Use of the Ada Programming Language in High Integrity Systems" [7] does not define a subset but identifies the basis on which a subset might be selected. The general approach is to identify verification techniques, then compare each technique with Ada language features and assess the level of compatibility.

Particular points of interest are exceptions and tasking. Exceptions are *essential* for high-integrity applications but must be *avoided* because of the difficulties they cause for flow and symbolic analysis. HRG had to reconcile these views; propagation of exceptions is clearly a key issue. For tasking they chose to define the "Ravenscar Profile", of which more later.

3.6 SPARK and other Ada subsets

SPARK [8] is a sub-language of Ada with particular properties that suit it to the most critical applications:

- completely unambiguous;
- free from implementation dependencies;
- all rule violations are detectable;
- formally defined and tool supported.

The base languages are ANSI/MIL-STD-1815A-1983 (SPARK 83) and ISO-8652;1995 (SPARK 95). SPARK removes the troublesome language features (e.g. tasking, access types), limits the way remaining features may be used (e.g. limitations on the placement of exit and return) and introduces annotations to provide extra information.

SPARK annotations describe program design information. They are related to executable code by static-semantic rules, which are checked mechanically by the SPARK Examiner tool. This tool also checks language subset compliance, does system-wide data flow and information flow analysis, formal verification including proof of safety and security properties, and can demonstrate prior to execution that a program is "exception free". SPARK is compatible with both HRG guidance and compiler-defined high-integrity subsets.

Other Ada subsets tend to fall into two groups: informal coding standards involving fairly arbitrary exclusion of language features, such as Systeam "Safe Ada", and subsets generated by compiler back-end and run-time library considerations. The latter are exemplified by GNAT Pro High Integrity Edition, C-Smart and Raven.

4 An overview of standards

There are many standards relating to software development for safety-related or security-related systems. Each domain (e.g. rail, aerospace, automotive) has its own standards. They vary in their power to mandate or recommend particular practices. In this section we discuss some representative standards and their implications for software development practice.

4.1 Software development techniques

Common aspects of standards' recommended or mandated techniques for software development include:

- hazard or risk analysis;
- defining safety integrity levels;
- defining the system lifecycle;
- formal methods for requirements and design;
- modelling various system properties;
- choice of language, RTOS etc.;
- validation, verification and testing (VVT); and
- accumulation of evidence in a safety case.

We now examine selected techniques in more detail.

4.2 Hazard analysis

Hazard analysis is the process of identifying hazards and their causes. A *hazard* is defined by Leveson [9] as "a certain state of a system that, combined with other conditions in the environment, will lead inevitably to an accident or loss". An example of a hazard for a car is "the wheels fall off while the car is travelling at speed". Each hazard in a system has an estimated probability of occurrence and an associated severity.

Associated terms are *risk* and *safety*. *Risk* is a combination of probability and severity: the above hazard is severe, but (hopefully) very unlikely in most modern cars, so does not carry great risk. There are three main "risk regions": broadly acceptable, As Low As Reasonably Practical (known as ALARP, of which more later) and intolerable. *Safety* is freedom from unacceptable risk.

Various formalised hazard analysis techniques exist, such as Hazard and Operability (HAZOP), Failure Modes and Effects Analysis (FMEA) and Fault Tree Analysis. Their common theme is the creation and maintenance of a hazard log, detailing the identified hazards and stating what steps have been taken to mitigate them.

4.3 Risk and ALARP

Risk itself is regulated (in the UK) on the basis of being reduced As Low As Reasonably Practical (ALARP). This stems from a UK Court of Appeal decision on the 1949 case Edwards vs. The National Coal Board where Judge Asquith noted:

"...a computation must be made by the owner in which the quantum of risk is placed on one scale and the sacrifice involved in the measures necessary for averting the risk (whether in money, time or trouble) is placed in the other, and that, if it be shown that there is a gross disproportion between them - the risk being insignificant in relation to the sacrifice - the defendants discharge the onus on them." [10]

4.4 Safety integrity levels

It is common when developing against a safety standard for the system to be assigned a *safety integrity level* (SIL), typically in the range 1 (low) to 4 (high). The techniques (and costs) for system validation will be defined by the standard for each SIL.

A SIL is a general indicator of the quality required for design / build processes. It is applicable to both software and hardware, and indicates the probability that the system meets its requirements and avoids systematic failure. However, it is often misunderstood and misused.

It is important to note that if software is developed to e.g. SIL 4 for a particular system then it is *not* automatically SIL 4 for a different system. The hazards against which the software was originally may not include all the hazards of the new system, so although the software might be high quality it has not been shown to be safe or secure in the new environment.

4.5 The safety case

A safety case is a collection of evidence for safety of a system in the form of an argument for overall safety. It will relate various verification and analysis activities to avoidance or mitigation of the system hazards. Typically a safety case may contain or reference:

- the configuration management plan
- the software verification plan
- the software quality assurance plan
- standards for design and coding
- specifications for requirements, design and tests
- the results of software verification
- a software configuration index; and
- a traceability matrix.

5 DO-178B

RTCA DO-178B "Software Considerations in Airborne Systems and Equipment Certification" [11] is a set of guidelines produced by the civil avionics industry for good practice in producing avionics software. It is neither a process document nor a standard, but in practice it has been endorsed by the FAA and JAA and is regarded as the "Bible" for civil avionics software.

DO-178B defines five levels of criticality for software based on consequence of failure, from E (no effect) through to A (catastrophic) in a similar approach to SILs. Only levels A and B are regarded as safety-critical.

5.1 Qualification and verification

For the output of each development step, the standard requires that either the tool that produced that output (e.g. the compiler) be *qualified*, or the output itself (e.g. object code) be *verified*. Since most tools in the object generation path are not qualified, the object code itself must be verified. This is achieved by specified test coverage criteria according to the criticality level. The coverage can be of the source code rather than the object code if source-to-object traceability can be achieved.

5.2 Coverage

In modified decision/condition branch coverage (MC/DC) testing, each value independently affecting a decision must be executed. In all cases the guidelines require coverage analysis to show that the coverage required has been attained. At Level A, the coverage must take place on the object code if the object code is not *directly traceable* from the source code – if Ada run-time checks are turned on in the compiler, for instance, then it is unlikely that such traceability exists.

Note that it is often possible to "cheat" on coverage. One way of reducing the number of tests required in Ada is to translate an if-else if – else block into a single lookup into a constant array, reducing many tests (one for each branch, for instance) to a single test – there is no concept of "covering" the data in such an array. For this reason, and others, coverage should not be regarded as a gold standard.

5.3 DO-178B Critique

DO-178B is a software correctness standard, not a system safety standard; there is no relation of the software to the system hazards, the developer can only state "the whole box has been tested to level A".

There is undue emphasis on testing activities in the document. This may reflect its age (released in 1992) but means that there is little or no credit for analysis and review which are both worthwhile activities. It can create an environment where getting to the testing phase quickly is seen as the prime aim, and where MC/DC is widely misinterpreted as "exhaustive" testing when we have seen how it can be circumvented to some extent.

The role of the Designated Engineering Representative (DER) is a specialist designated by the FAA is to establish compliance with the DO-178B process, not system safety. In this respect DO-178B is very different from more modern safety-related standards.

5.4 Economic arguments

The cost of testing to DO-178B rises as the criticality level rises. Most testing work needs a test rig, the development of which is a back-end high-risk process. MC/DC requires many more tests than statement coverage; one estimate is that it multiplies the cost of testing by 5.

As noted, the required testing for level A is likely to be harder in Ada than in C. Is the best solution therefore to "hack in C"? No – this is because it is much cheaper to adopt a process that reduces errors *before* going into formal test. This was the approach taken when developing software for the C-130J Hercules II aircraft.

5.5 C-130J development

The C-130J was a project by Lockheed Martin which aimed to produce a much-improved version of the venerable C-130 Hercules transport aircraft. It included a new propulsion system with 29% more thrust and 15% better fuel efficiency, advanced avionics and control systems, two mission computers and two backup bus interface units to give dual redundancy to the aircraft systems.

The development process emphasised "Correctness by Construction" (CbC), focused on "doing it right first time". The requirements of DO-178B were kept in mind but not allowed to dominate the development. Development was driven by verification, aiming to verify objects as soon as they were created.

Formality was introduced early, in the specification phase (using the CoRE requirements engineering process and Parnas table to specify in/out mappings) and in the programming language (Ada 83 and SPARK). There was a close map from the requirements to the code: "one procedure per table". Static analysis (with the SPARK Examiner) was part of the development process

5.6 C-130J results

There was an unexpected benefit of using SPARK; the language requires all data interactions to be described in annotations, but some interactions were not clearly specified in the CoRE tables (e.g. validity flags). The coders could not ignore the problem, and so the requirements/specification document became a dynamic document as ambiguities were resolved during coding.

Early static analysis picked up a number of errors which would have been unlikely to surface during testing. The testing required by DO-178B was greatly simplified as a result. Very few errors were found during testing, which was done for less than 20% of the normal industry cost, and the overall cost of the level A system was half that of typical non-critical system development cost.

5.7 C-130J UK verification

The UK Ministry of Defence commissioned Aerosystems International to perform retrospective static analysis of *all* the C-130J critical software, using a variety of tools e.g. the SPARK Examiner and SPADE toolset for proving SPARK code against CoRE, and MALPAS for Ada. All anomalies were investigated and "sentenced" by system safety experts. The results were instructive:

- significant safety-critical errors were found by static analysis in code developed to DO-178B Level A;
- proof of SPARK code was shown to be cheaper than other forms of static analysis performed;
- SPARK code had only 10% of the residual errors of full Ada, and Ada only 10% of the residual errors of C; and
- there was no statistically significant difference in residual error rate between DO-178B Level A, Level B and Level C code.

The results of the C-130J development are detailed by Sutton and Croxford [12] and Amey [13] and compared by German [14] to the results of other development methods.

6 UK Defence Standards

The UK Ministry of Defence uses Defence Standard 00-56 Issue 2 [15] as its primary safety standard. This references Defence Standard 00-55 for specific requirements and guidance on developing safety-related software. 00-56 is a system-wide standard and defines the safety analysis process to be used to determine safety requirements, including SILs. It promotes an active, managed approach to safety and risk management.

6.1 Defence Standard 00-55

00-55 Issue 2 is applied to software components of various SILs. It defines the process and techniques to be followed to achieve a level of rigour appropriate to the SIL. It requires the production of a software safety case to:

- justify the suitability of the development process, tools and methods; and
- present a justification based on objective evidence that the software satisfies the safety aspects of the software requirement.

Compared to DO-178B, 00-55 puts a heavy emphasis on the use of formal methods and proof. It still requires stringent testing. It uses integrity levels S1 to S4 rather than criticality levels E to A, but the main difference is that 00-55 focuses on software safety whereas DO-178B focuses on software correctness.

6.2 SHOLIS – a 00-55 project

The Ship Helicopter Operating Limits Instrumentation System (SHOLIS) is a system for assessing and advising on the safety of helicopter flying operations on board Royal Navy and Royal Fleet Auxiliary vessels. Power Magnetics and Electronic Systems Ltd. was the prime contractor. Praxis Critical Systems developed all the operational software to *Interim* Defence Standard 00-55 (1991).

SHOLIS is a dual redundant system, implementing fault tolerance by hot-standby. The two data processing units (DPUs) are located at opposite ends of the ship to increase tolerance of battle damage, and are not synchronised in any way. It was the first system with software developed to Interim 00-55 at S4.

The final requirements numbered 4000 statements, with 300 pages of Z, English and mathematics. The code was 27,000 lines of non-blank non-comment Ada plus 54,000 lines of SPARK flow annotations and 20,000 lines of SPARK proof annotations.

6.3 SHOLIS verification

The major verification activities were:

- review and Z proof of the System Requirements and System Design;
- SPARK static analysis and proof of the code;
- worst-case timing, stack-use and I/O estimates;
- module, integration and system validation tests based on Z specifications and requirements;
- testing to cover 100% statement and MC/DC;
- acceptance, endurance and performance testing; and
- review of everything with a documentation trail.

As of January 2001 the system had passed all system validation, endurance and performance tests. The acceptance test for the customer was completed with no problems, and following sea trials some cosmetic changes were requested but no faults, crashes or unexpected behaviour were reported.

6.4 SHOLIS conclusions

Z proof and system validation tests were the most cost-effective phases at finding faults. Traditional module testing was arduous and found few faults, except in fixed-point numerical code. The strong static analysis removed many common faults before they could be tested. Software integration work was trivial. More detailed conclusions and fault metrics are given by King et al. [16].

The proof work demonstrated proof of exception freedom on the whole system, which was a significant win. In the proof work, the major lesson at the time (1998) was that a big computer is far cheaper than the time of the engineers using it. Moving on to today, the proof simplification work would be better done distributed on personal workstations, reducing the original simplification time by 1-2 orders of magnitude.

The conclusion of the SHOLIS development work is that the use of formal techniques such as proof and static analysis not only satisfies the stringent requirements of UK defence standards, but provides a real payback in terms of reduced testing time and increased system assurance.

6.5 00-56 revision

Defence Standard 00-56 is being revised, and the second public draft for comments was released in January 2004. The new version incorporates Defence Standards 00-55 and 00-54 (safety-related electronic hardware) merged under the title “Programmable Systems”. We will briefly examine the new approach in this draft standard, mindful that it may change before release as Issue 3.

The approach in the new 00-56 is to require the developer to provide evidence that the system is safe for its intended purpose throughout its life. This will include a safety case, an auditable safety management system, risk assessment, hazard analysis, and a demonstration that risk is tolerable or ALARP. The developer must define the safety standards that they will meet, and then demonstrate compliance.

The Code of Practice in 00-56 refers to the software as a Safety Related Programmable System (SRPS). It requires *validated* safety arguments or claims for the SRPS, supported by:

- direct evidence from deductive analysis, demonstration and review;
- process evidence from following good practice; and
- qualitative evidence from good design.

Tools and processes which might undermine the SRPS integrity (e.g. compilers or analysis tools) must be qualified in some way. The standard also explicitly requires consideration of counter-evidence i.e. testing and in-service failures.

7. The Common Criteria

Security-critical applications are as important as safety-critical applications. We now consider the international Common Criteria security evaluation guidelines, and see how a security application (the Mondex Certification Authority) was developed and verified.

7.1 Common Criteria concepts

The Common Criteria for IT Security Evaluation [17] aims to set a “level playing field” for all developers in the participating states, and aims for international mutual recognition of evaluated products. It combines and replaces the US “Orange Book” and the UK ITSEC scheme.

The Common Criteria (CC) defines two types of IT security requirement:

- functional (defining the behaviour of a system or product); and
- assurance (for establishing confidence in the implemented security functions – is the product built well and does it meet its requirements?)

The CC defines seven Evaluation Assurance Levels – EAL 1 (lowest) through EAL 7 (highest). An EAL defines a set of functional and assurance components from the CC documents which must be met. EAL 7 roughly corresponds to ITSEC E6 and Orange Book A1.

7.2 The MULTOS CA

MULTOS is a multi-application operating system for smart cards. It allows applications to be loaded and deleted dynamically when the card is “in the field”. To prevent forging or the use of invalid applications, applications and card-enabling data are signed by the MULTOS Certification Authority (CA). At the heart of the CA is a high-security computer system that issues these certificates.

The CA has some unusual requirements:

- a target of 6 months between reboots and warm stand-by fault tolerance;

- a lifetime of decades, and must be supported for that long;
- to be tamper-proof and subject to the most stringent physical and procedural security; and
- meet the requirements of UK ITSEC E6.

All the requirements, design, implementation and on-going support for the CA were done by Praxis Critical Systems.

7.3 MULTOS CA development process

The overall development process conformed to ITSEC E6, minimising the reliance on COTS by assuming arbitrary but non-Byzantine behaviour. The COTS components (e.g. Windows NT, Backup tool and SQL Server) were not certified.

The security policy and top-level specification was formalised in Z, although no formal proof was carried out. The system was implemented in a mix of SPARK Ada, full Ada 95, Visual C++ and SQL, and tested top-down with coverage measurement. The mixed-language approach was due to selecting “the right tools for the right job” – implementing a GUI in SPARK Ada was clearly as wrong as implementing security-critical functions in C++.

The use of SPARK in particular was because it is almost certainly the only industrial-strength language that meets the requirements of ITSEC E6. Full Ada 95 was used where the necessary constructs (e.g. tasking) were not in the SPARK language at the time; even then, the use of Ada was “Ravenscar-like” in using simple, static allocation of memory and tasks.

7.4 MULTOS CA results

The use of Z for the formal security policy and system specification helped to produce an indisputable specification of functionality, and using Z, CSP and SPARK “extended” the formality into the design and implementation. The top-down incremental approach to integration and test was effective and economic. High-security systems incorporating COTS are possible, and indeed probably necessary.

Note that the CA was not formally evaluated against ITSEC E6; nevertheless, there was a clear benefit in security and reliability from developing to this standard.

8 Compilers and runtime

Having established and demonstrated approaches to developing high-integrity systems for the safety and security domains we now examine how developers should select a compiler and run-time for such a high-integrity system.

8.1 High-integrity compilers

A high-integrity Ada compiler should have:

- Annex H support;
- evidence of qualification against the Ada standard;
- optimisation and other switches;
- available and competent support;
- runtime support for high-integrity systems; and
- support for object code verification.

The HRG report [7] recommends validation of appropriate annexes – almost certainly A, B, C, D and H. Annex G (Numerics) may also be applicable for some systems. It does *not* recommend use of a subset compiler, although it recognises that a compiler may have a mode in which a particular subset is enforced. The main compiler algorithms should be unchanged in such a mode.

8.2 Compiler features

Annex H support and available pragmas will vary among compilers. One of the most important considerations is whether the compiler vendor has documented implementation decisions – demand this information from the vendor, and if they cannot or will not supply such information then find out why not!

“Qualification” of a full compiler as per the DO-178B (or similar) definition is beyond our reach at the moment. The pragmatic alternative is to combine:

- avoidance of “difficult to compile” language features in the high-integrity subset used;
- in-service history of the compiler;
- choosing the “most commonly used” compiler options;
- a study of the compiler faults history and database;
- verification and validation of the system; and
- object code verification as a last resort.

8.3 Runtime systems

In delivering a high-integrity system, a run-time library (RTL) must be verified to the same level of assurance as the rest of the application. Most Ada compiler vendors have responded to this need with products for Ada 83 and Ada 95. There are three main approaches to certifiable runtime systems: “small and certifiable”, Ravenscar or no runtime.

The “small and certifiable” approach is largely aimed at meeting the requirements of DO-178B up to and including level A systems. The technical goal is to eliminate language features with an unacceptably large runtime impact, e.g. tasking, heap allocation, exceptions, predefined I/O. Examples are Aonix C-SMART and Rational VADSsc for Ada 83.

Ravenscar is a “profile” of tasking and other features in Ada 95 which is appropriate for high-integrity and hard real-time systems. It is designed to be a particularly efficient, simple runtime system implementation on a single processor target, amenable to static timing and schedulability analysis. An example implementation is Aonix Object Ada Real-Time/Raven. The SPARK subset now includes Ravenscar.

The idea of "no runtime" is to eliminate all language features which require any runtime library support. The advantage is that there is no COTS component to certify.

8.4 Runtime options and development

For all runtime options, the Board Support Package is necessary. This provides support for download and start up, cold boot (i.e. from ROM), hardware-specific initialisation, debugging, coverage analysis and some predefined simple I/O.

The BSP must be tested, but traditional testing techniques such as unit test and coverage analysis may not work on the BSP. In some projects we have fielded two BSPs: a "debug" version supporting all the above functionality for *any* application, and a "strip" version supporting only the functionality required for delivery of a specific application. This makes (manual) coverage analysis and testing easier, with no "dead" code.

8.5 Object code verification

If a compiler cannot be trusted sufficiently, manual verification of the object code may be required (OCV). This should be avoided if at all possible as it requires detailed knowledge of the language, compiler and target processor, and is hard and lengthy work.

The 1-step approach to OCV is to compare source code with disassembled object code directly. However the "semantic gap" between the two is very wide, especially for a rich language like Ada, and this therefore requires detailed knowledge of Ada compilation, code generation, language issues etc.

The 2-step approach is to review the Ada source against the compiler-generated intermediate language (IL), then review the IL against the disassembled object code. This means that the semantic gaps are narrower and splits the reviewing skills required, but not many compilers allow users access to IL.

From experience on the SHOLIS project, where a compiler feature caused large aggregates to be allocated on the heap although no heap was used in the runtime, we recommend always having someone on a project team who is capable of reading and reviewing the object code.

9 Conclusions

In this article we have examined the challenge of implementing high-integrity systems in the context of our experience with them. We have shown what the current safety and security standards demand, described three real-world commercial projects developed under these standards, and shown how the selection of tools and development processes can pay off in reduced development time and increased system reliability.

References

[4] B. Littlewood and L. Strigini (1993), *Validation of Ultrahigh Dependability for Software-based Systems*, Communications of the ACM, November 1993.

[5] R. W. Butler and G. B. Finelli (1993), *The Infeasibility of Qualifying the Reliability of Life-critical Real-time Software*, IEEE Transactions on Software Engineering, vol. 19 no. 1, January 1993, pp. 3—12.

[6] B. A. Wichmann (1992), *Requirements for Programming Languages, Computer Standards and Interfaces*, National Physical Laboratory.

[7] ISO/IEC JTC1/SC22/WG9, *Programming Languages – Guide for the Use of the Ada Programming Language in High Integrity Systems*, http://www.dkuug.dk/jtc1/sc22/wg9/n359.pdf

[8] J. Barnes (2003) *High Integrity Ada – the SPARK Approach to Safety and Security*, Addison Wesley, ISBN 0-321-13616-0

[9] N. G. Leveson (1995) *Safeware: System Safety and Computers*, Addison-Wesley, ISBN 0-201-11972-2

[10] Judge Asquith (1949) *Edwards v. National Coal Board*, All England Law Report Vol. 1, p. 747

[11] RTCA (1992) *Software Considerations in Airborne Systems and Equipment Certification*, DO-178B

[12] J. Sutton and M. Croxford (1996) *Breaking Through the V&V Bottleneck*, Lecture Notes in Computer Science vol. 1031, Springer-Verlag

[13] P. Amey (2002) *Correctness by Construction: Better Can Also Be Cheaper*, CrossTalk journal, March 2002

[14] A. German (2003) *Software Static Code Analysis Lessons Learned*, CrossTalk journal, November 2003

[15] UK Ministry of Defence (1996) *Safety Management Requirements for Defence Systems*, Defence Standard 00-56, Issue 2

[16] S. King, R. Chapman, J. Hammond and A. Pryor (2000) *Is Proof More Cost-Effective Than Testing?* IEEE Transactions on Software Engineering, vol. 26 no. 8, August 2000

[17] National Institute of Standards and Technology (1999) *Common Criteria for Information Technology Security Evaluation*, CCIMB-99-031, version 2.1, August 1999

No Pointers, Great Programs

Mário Amado Alves

University of Porto, Rua do Campo Alegre 823, 4150-180 Porto, Portugal; email: maa@liacc.up.pt

Abstract

This article summarizes the half day tutorial given at RST 2004, fully entitled "No Pointers, Great Programs. How to stay on the value semantics side of the Ada way with a little help from containers."

1 The Ada ways

In this half day tutorial at RST 2004 [1], we explore the value semantics side of the Ada way. The tutorial is targeted mainly at beginners and newcomers to Ada, so we review the Ada foundations for value semantics, including unconstrained types, parameter modes, and class-wide types.

The Ada way is contrasted with the ways of C-like languages, and the 'new' value semantics way is contrasted with the 'traditional' Ada way regarding certain constructions e.g. heterogeneous arrays and ragged arrays. Whereas the traditional way uses pointers (access types and values), the new way uses containers.

2 The inescapable string examples

To contrast with the C way, the paradigmatic example of passing a string is used:

```
do_something (char * s);
Do_Something (S : String); -- no pointers!
```

Strings continue to be used to introduce the container concept, namely with Unbounded_String. The memory magic of Unbounded_String is focused upon:

> *No storage associated with an Unbounded_String object shall be lost upon assignment or scope exit.* §A.4.5 (88)

(The notation § is used to refer to parts of the Reference Manual [2].)

As an example of a fitting use of Unbounded_String, a function Escape is specified that prefixes a backslash to any special character in a string:

```
function Escape (Source : String) return String is
  U : Unbounded_String;
begin
  for I in Source'Range loop
    if Is_Special (Source (I)) then Append (U, '\'); end if;
    Append (U, Source (I));
  end loop;
  return To_String (U);
end;
```

Ragged string arrays, i.e. one-dimensional arrays whose components are strings of possibly different sizes, are used to contrast the traditional way, using pointers:

```
type String_Ptr is access String;

function "+" (S : String) return String_Ptr is
begin
  return new String'(S);
end;

Last_Names : array (Positive range <>) of String_Ptr :=
  (+"Eachus",
   +"Leif",
   +"Kasakov");
```

with the new, pointerless way, using the Unbounded_String container:

```
function "+" (S : String) return Unbounded_String
  renames To_Unbounded_String;

Last_Names : array (Positive range <>) of Unbounded_String :=
  ...
```

Other composite types, namely records, and other containers, are also studied.

3 Cells

The cell abstraction is introduced as an alternative to pointers. A cell encapsulates a possibly indefinite type inside a definite one, the cell container.

```
generic

  type Element_Type (<>) is private;

package Cells is

  type Cell_Type is private;

  function Put (Item : Element_Type) return Cell_Type;
  function Get (Cell : Cell_Type) return Element_Type;
```

An abstraction of simple arithmetic expressions is given as an example of a fitting use of Cells, together with class-wide programming. An expression is a recursive structure:

```
expression ::= integer | operation (expression, expression)
operation ::= add | multiply
```

The traditional way must use pointers:

```
type Expression is abstract tagged null record;

function Value (X : Expression) return Integer is abstract;

type Expression_Class_Ptr is access all Expression'Class;

type Operation is abstract new Expression with
  record
    Left_Operand : Expression_Class_Ptr;
    Right_Operand : Expression_Class_Ptr;
  end record;
...
```

The pointerless way uses containers:

```
package Expression_Cells is new Cells (Expression'Class);
use Expression_Cells;

type Operation is abstract new Expression with
  record
    Left_Operand : Cell_Type;
    Right_Operand : Cell_Type;
  end record;
```

The tutorial notes include the full source code of all examples.

4 Ada.Containers

The forthcoming standard container library for Ada 2005 is overviewed:

Definite element type	*Indefinite element type*
Vectors	Indefinite_Vectors
Doubly_Linked_Lists	Indefinite_Doubly_Linked_Lists
Hashed_Maps	Indefinite_Hashed_Maps
Ordered_Sets	Indefinite_Ordered_Sets

An already existing reference implementation is identified and used: AI302, mantained by Matthew Heaney at Tigris.org.

Several examples of use are presented, including an implementation of Cells based on Indefinite_Vectors.

5 Conclusions

A practical application, XML_Translator, is presented as an example of building complex data structures the pointerless way.

A drawback of this way is noted: for complex, and particularly *cyclic*, structures, strict value semantics force the copy of the entire structure upon each update of an element, which hinders efficiency for large structures.

Finally, the following conclusions are drawn:

Value semantics approach: no pointers.

Mandatory for small values.

Excellent for *inspection* of structures of any size and complexity.

Good for *change* of simple structures of any size (because of by reference passing).

Problems with *many changes* to large complex structures.

When pointers necessary:

Use *smart*, or *safe*, pointers (another tutorial).

In any case: use Ada!

100% Ada, 0% bugs.

6 Miscellaneous information

Tutorial notes website (most up-to-date):

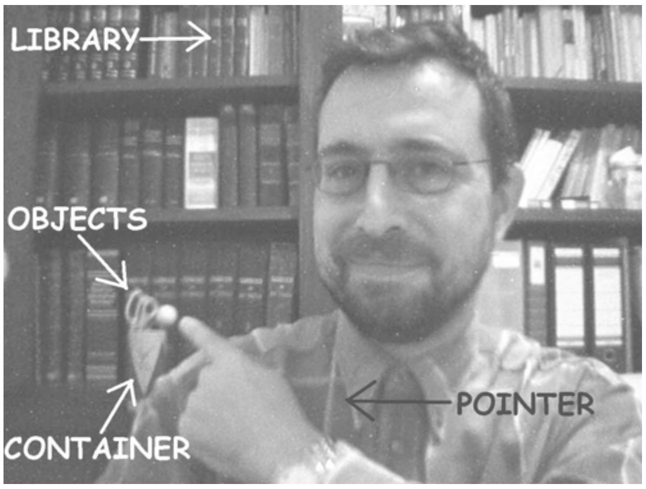
http://www.liacc.up.pt/~maa/no_pointers

Tutorial data sheet at the RST 2004 website:

http://dmi.uib.es/~AE2004/documents/tutorials/tutorial04.pdf

About the presenter

M. A. Alves is currently with the University of Porto, doing PhD research on adaptive hypertext. He has degrees in Linguistics and Informatics Engineering, and has taught Ada, Software Engineering and other courses at several universities and institutions. He has been a software developer since the mid 1980's and an Adaist since the mid 1990's. He has been in the Ada Standard Container Library Working Group since its inception in the RST 2002 workshop. Lately he has contributed to Ada Issue 302 for the creation of such a library for Ada 2005, where he has focused on indefinite elements and persistence. His latest persistent container experiments include Mneson, a developing 100% Ada database system, available from his homepage at http://www.liacc.up.pt/~maa



References

[18] A. Llamosi and A. Strohmeier (eds) (2004), *Reliable Software Technologies Ada-Europe 2004*, LNCS 3063, Springer-Verlag.

[19] T. S. Taft et al. (eds) (2001), Consolidated Ada Reference Manual, LNCS 2219, Springer-Verlag (online at www.adaic.org)

Ada-Europe 2004 Sponsors

ACT Europe
Contact: Zépur Blot

8 Rue de Milan, F-75009 Paris, France
Tel: +33-1-49-70-67-16 Fax: +33-1-49-70-05-52
Email: sales@act-europe.fr URL: www.act-europe.fr

Aonix
Contact: Anne Chapey

66/68, Avenue Pierre Brossolette, 92247 Malakoff, France
Tel: +33-1-41-48-10-10 Fax: +33-1-41-48-10-20
Email : info@aonix.fr URL : www.aonix.fr

Artisan Software Tools Ltd
Contact: Emma Allen

Suite 701, Eagle Tower, Montpellier Drive, Cheltenham, GL50 1TA, UK
Tel: +44-1242-229300 Fax: +44-1242-229301
Email : info.uk@artisansw.com URL : www.artisansw.com

Green Hills Software Ltd
Contact: Christopher Smith

Dolphin House, St Peter Street, Winchester, Hampshire, SO23 8BW, UK
Tel: +44-1962-829820 Fax: +44-1962-890300
Email : chriss@ghs.com URL : www.ghs.com

I-Logix
Contact: Martin Stacey

1 Cornbrash Park, Bumpers Way, Chippenham, Wiltshire, SN14 6RA, UK
Tel: +44-1249-467-600 Fax: +44-1249-467-610
Email : info_euro@ilogix.com URL : www.ilogix.com

LDRA Ltd
Contact: Jim Kelly

24 Newtown Road, Newbury, Berkshire, RG14 7BN, UK
Tel: +44-1635-528-828 Fax: +44-1635-528-657
Email: info@ldra.com URL: www.ldra.com

Praxis Critical Systems Ltd
Contact: Rod Chapman

20 Manvers Street, Bath, BA1 1PX, UK
Tel: +44-1225-823763 Fax: +44-1225-469006
Email : sparkinfo@praxis-cs.co.uk URL : www.sparkada.com

Scientific Toolworks Inc
Contact: Matthew Bergeson

321 N. Mall Drive Suite I-201, St. George, UT 84790, USA
Tel: +1-435-627-2529 Fax: +1-877-512-0765
Email: sales@scitools.com URL: www.scitools.com

TNI Europe Limited
Contact: Pam Flood

Triad House, Mountbatten Court, Worrall Street, Congleton, Cheshire CW12 1DT, UK
Tel: +44-1260-29-14-49 Fax: +44-1260-29-14-49
Email: info@tni-europe.com URL: www.tni-europe.com