ADA USER JOURNAL

Volume 22
Number 4
December 2001

Contents

Editorial Policy for *Ada User Journal*

Publication

Ada User Journal – The Journal for the international Ada Community – is published jointly by Ada Language UK Ltd and 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 Language UK Ltd and Ada-Europe an unlimited licence 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Unless the authors state explicitly otherwise, submission of an article is taken to imply that it represents original, unpublished work, not under consideration for publication elsewhere.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 Language UK Ltd, Ada-Europe or their 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. Electronic submission is preferred – typed manuscripts will only be accepted by the Editor 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 at:

www.adauk.org.uk

There is no limitation on the length of papers, though a paper longer than 10,000 words would be regarded as exceptional.

Editorial

Once more, the news section contains many interesting items. Ada success stories always make pleasant reading, for example, see the Ada Inside section of the news with regards to the Joint Strike Fighter aircraft. The ongoing “Ada versus C++ versus Java” debate features heavily in the Ada in Context news section, with interesting arguments from all perspectives.

The technical article in this issue provides an overview of SPARK Ada. The article illustrates how the use of abstraction within a SPARK program leads towards proof of correctness. A number of useful examples are given.

Within the Forthcoming Events section are further details of the Ada-Europe 2002 conference. Invited speakers include Maarten Boasson, Alois Ferscha, Rachid Guerraoui and Mehdi Jazayeri. Tutorial sessions are prominent in the conference, including sessions on software design and patterns in Ada, SPARK Ada, software testing and metrics.

Finally, I draw your attention to the first article in the news section regarding the Embedded Systems Club. This new organisation is offering free CDROM versions of the Embedded Systems Resource Library to the first 500 to register as associate members.

Neil Audsley
York
December 2001
Email: Neil.Audsley@cs.york.ac.uk

News

Dirk Craeynest (ed)

Offis nv/sa and K U Leuven. Email Dirk.Craeynest@offis.be

Contents

Ada-related Organizations

The Embedded Systems Club

From: John@Adaxia.com
Date: Mon, 22 Oct 2001 10:10:30 +0100
Organization: John Robinson & Associates
Subject: The Embedded Systems Club
Newsgroups: comp.lang.ada

The Embedded Systems Club has been created to improve communication among members of the embedded systems community.

The club organises conferences, distributes the "Embedded Systems Resource Library" on CD, maintains a club web site and facilitates the formation of Special Interest Groups (SIGS) on embedded systems issues.

Associate Membership is available now and is free of charge. The first 500 people to register as associate members will receive a copy of the first release of the resource library.

More details are available at: http://www.EmbeddedSystemsClub.com or from the club manager: mailto:Hazel@Adaxia.com

The Embedded Systems Club is sponsored by: Aonix Europe, http://www.aonix.co.uk; Artisan Software, http://www.artisansw.com; I-Logix, http://www.ilogix.com; John Robinson And Associates, http://www.JohnRobinsonAndAssociates.com

Ada-related Events

Upcoming Ada-related Conferences

From: Clyde Roby <roby@ida.org>
Date: Fri, 2 Nov 2001 08:01:22 -0500
Subject: ABWG and Conferences reminder
To: SIGAda-ABWG@acm.org

Message for November 2001:

Don't forget about our upcoming Ada-related conferences:

- o 11th International Real-Time Applications Workshop (IRTAW 11) 9-12 April 2002, Mont-Tremblant, Quebec, Canada
- o Ada-Europe'2002 -- 7th International Conference on Reliable Software Technologies: Vienna, Austria, 17-21 June 2002 http://www.ada-europe.org/conference2002.html
- o SIGAda 2002 -- Houston, Texas, USA (dates TBD) -- http://www.acm.org/sigada/conf/sigada2002

Ada-Belgium Announces Ada Programming Competition

From: dirk@cs.kuleuven.ac.be (Dirk Craeynest)
Date: 7 Nov 2001 23:18:45 +0100
Organization: Ada-Belgium, c/o Dept. of Computer Science, K.U.Leuven
Subject: Ada-Belgium announces Ada programming competition
Newsgroups: comp.lang.ada,be.comp.programming

Ada-Belgium offers 400 Euro for the best example of what Ada can do.

http://www.cs.kuleuven.ac.be/~dirk/ada-belgium/contest/

Ada-Belgium announces a competition open to all. We'd like to see examples of what Ada can do. We'll award a prize of 400 Euro to the entry we judge to be the best.

We're looking for an interesting idea and good Ada. The judges' decision will be based on originality, utility, quality, style, reusability, readability and good use of Ada features.

You can enter existing work or new material written for the contest. We will consider all types of projects: applications, libraries or even incomplete works. If you are building on top of an existing library or other piece of code then your work should be indicated clearly and should predominantly be in Ada; the library you use need not be. The judges will consider the code itself; you should indicate the most interesting parts of your source - about 1K lines - for the judges to examine. If you are targetting a reasonably common computer and using generally available tools, then you are welcome to supply build instructions so the judges can try to execute your code. If your target is portable you may be able to demonstrate the code running.

Judging will take place in January and February. To help plan the work of the judges we'd like you to register your interest in advance. The competition is open to everyone. Entries must be submitted in one of English, French, Dutch or German. Please register as soon as possible using the contact information below.

Your entry should reach us by the end of the seventh of January, 2002. We would like to publish the best entries. Please make clear when you submit the entry whether it may be published and under what terms. By default the GNU Public Licence will be assumed if none is mentioned.

The judges will be members of or appointed by the Ada-Belgium board. The judges decisions are final. The judges will not enter into correspondence regarding their decisions.

Ada-Belgium reserves the right to award additional prizes. Funds were already made available by the sponsors for this purpose.

Contact: ada-belgium-contest@cs.kuleuven.ac.be

Organized and sponsored by: Ada-Belgium. Additional sponsors: AdaPower.com.

Ada and Education

Ada Training

[This information is included as examples of public Ada training courses: many are being organized regularly. For more, see also "Ada Training" in AUJ 21.3 (October 2000), p.161. -- dc]

From: "Ed Colbert" <colbert@abssw.com>

Date: Sat, 22 Sep 2001 07:50:30 -0700
Organization: Absolute Software Co., Inc.
Subject: [Announcing] Public Ada 83 & Ada 95 Classes during October & December in Carlsbad CA
Newsgroups: comp.lang.ada

Absolute Software will be holding a public Ada 83 class on the week of 15 October and Ada 95 on the week of 10 December. Both will be held in Carlsbad, CA. You can find a full description and registration form on our web-site, www.abssw.com. Click the Public Courses button in the left margin. (We also offer classes on object-oriented methods and other object-oriented languages.)

If there is anything you'd like to discuss, please call, write, or send me E-mail.

From: rod@praxis-cs.co.uk (Rod Chapman)
Date: 29 Nov 2001 04:36:13 -0800
Subject: ANN: SPARK Training Courses for 2002
Newsgroups: comp.lang.ada

We're pleased to announce the dates of 2 up-coming "Software Engineering with SPARK" Training courses, to be held at Praxis Critical Systems' offices here in Bath.

Course 1: 21st - 24th January 2002
Course 2: 8th - 11th April 2002

In addition to the normal content, these courses will cover the new and improved features of Release 6.0 of the SPARK Toolset.

More details, a course flyer, and booking forms are available on www.sparkada.com, or email Fiona Joy at sparkinfo@praxis-cs.co.uk

Ada Tutor

From: john@prousa.net (John Herro)
Date: 4 Oct 2001 06:48:35 -0700
Subject: AdaTutor Has Moved
Newsgroups: comp.lang.ada

[See also AUJ 21.1 (April 2000), pp.20-21. -- dc]

The Ada Tutor Web site has moved to www.adatutor.com. Also, our E-mail address has changed to john [at] adatutor [dot] com. (The address from which this message is posted is just a spam trap.)

For those of you maintaining Web pages with pointers to Ada tutorials, please update your links.

Most of the people in this newsgroup already know Ada. We want to reiterate that we expect shareware registrations only from people who use AdaTutor to learn Ada. If you already know Ada, we don't expect you to register, but we would be honored to receive any comments you may have about our tutorial.

John J. Herro, Software Innovations Technology

Software Engineering: On the Right Track

From: John McCormick <mccormic@cs.uni.edu>
Date: Thu, 11 Oct 2001 08:09:56 -0500
Subject: Re: Ada Web Page
To: team-ada@acm.org

[...] I appreciate your including my 1996 Team Ada posting on the comparison of C and Ada in my real-time embedded systems course. A more detailed version was published in CrossTalk and can be found at http://www.stsc.hill.af.mil/crosstalk/2000/aug/mccormick.asp

And if anyone is interested in the current state of my model railroad lab, I have started a web site at http://www.cs.uni.edu/~mccormic/RealTime/

John W. McCormick, mccormick@cs.uni.edu, john.mccormick@acm.org Computer Science Department, University of Northern Iowa, Cedar Falls, IA 50614-0507, voice (319) 273-2618, fax (319) 273-7123

Ada-related Resources

Ada and Software Engineering Library

From: "David C. Hoos" <david.c.hoos.sr@ada95.com>
Date: Mon, 17 Sep 2001 08:34:43 -0500
Subject: Re: Public Ada Library
Newsgroups: comp.lang.ada

> I've tried to access the PAL at wuarchive, but it seems to have disappeared. Does anybody know if it's still on-line somewhere?

It's at http://unicoi.kennesaw.edu/ase/index.htm

[The Public Ada Library (PAL) has evolved into the Ada and Software Engineering Library (ASE). A full mirror is also available on the Ada-Belgium ftp server at ftp://ftp.cs.kuleuven.ac.be/pub/Ada-Belgium/cdrom/index.html -- dc]

ProgrammingPages.com

From: "ProgrammingPages.com" <webmaster@programmingpages.com>
Date: Sat, 22 Sep 2001 18:05:15 +0100
Organization: The University of York, UK
Subject: Ada websites Wanted
Newsgroups: comp.lang.ada

The ProgrammingPages.com is a new site still under development but will hopefully soon be fully functional. The site is hopefully going to be a directory/top sites list covering all aspects of computer programming.

Please take a look at the site, although it is still in its initial stages of development. The site can be found at http://www.programmingpages.com.

[...] If you have a programming related website I would be very grateful if you could add it to the database of sites. [...] I would like sites to be included from as many different, however obscure languages possible. For this reason if the language that your site is about is not listed then you have the ability to add the language. Thanks, all comments and suggestions are more than welcome!

Marcus Robinson

Ada-related Tools

Booch Components

From: Simon Wright <simon@pushface.org>
Date: Sun, 2 Sep 2001 19:34:49 +0100
Subject: Booch Components 20010819
To: team-ada@acm.org

This release has been uploaded to http://www.pushface.org/components/bc/ and is mirrored at http://www.adapower.net/booch/.

Features:

Began work on a case study.

Added a missing 'with abort' to a requeue in BC.Support.Synchronization.

Bounded Bags, Maps and Sets use a bounded hash table. This reduces the space requirement considerably and means that the Available function returns the correct value. Iteration is much faster.

Began re-indenting to the GNAT default (basically, 3 spaces standard indent, 2 spaces for continuations).

From: "Ehud Lamm" <mslamm@mscc.huji.ac.il>
Date: Sat, 8 Sep 2001 00:04:27 +0200
Organization: The Hebrew University of Jerusalem
Subject: Re: avl tree - booch components
Newsgroups: comp.lang.ada

[From a thread on how to instantiate some Booch components. -- dc]

> There is also now a "case study" which doesn't address instantiating Trees but does cover Collections. May be some help. http://www.pogner.demon.co.uk/components/bc/case-study.html

This looks very promising. It is going to be a real help for those starting out with the BC library.

AdaSL - Ada Structured Library

From: minyard@acm.org (Corey Minyard)
Date: Fri, 28 Sep 2001 03:51:54 GMT
Subject: AdaSL 1.3 released
Newsgroups: comp.lang.ada

I have released a new version of the Ada Structured Library I have written. I haven't changed any old stuff, only added new things. In particular, I have added:

* A telnet protocol handler - This provides a full implementation of telnet, along with some option processors and a stream-based version of telnet.
* An abstract file I/O package - IMHO, Ada needs some type of abstract file I/O package. For instance, I wanted to implement something on top of my telnet package that looked like a file, so that all the applications using it wouldn't generally have to care if it was a telnet connection, serial port, or console they were talking to. This is a lot like Ada.Text_IO and its subtending packages, so it's pretty flexible. This is something I would like to see added to the Ada core language (Hint, Hint).
* A debug output framework - Applications often need a way to generate debug I/O when necessary, and be able to turn the debug output on and of by command. This provides a framework for doing that.
* A string tokenizer - Much like java.util.StringTokenizer, this provides a way to take a string and chop it into tokens.
* An interactive command processor - This provides a way to allow commands to be bound into a command processor, then executed when the user types that command. A full telnet implementation of this exists, it ties the debug output framework in for a complete application framework for debugging. It also has an optional security binding. This can be instantiated with just a few lines of code.
* Lots of little helpers to tie all these together, along with tests and some examples.

Since I'm not working, I've had some time to play with this. I'm hoping it's useful for people, and I'm hoping that things like this will help Ada succeed in the marketplace.

Oh, BTW, it's on SourceForge, you can get to it at http://adasl.sourceforge.net

I consider this release somewhat beta, and I'll be glad to take comments on improvements, bug fixes, or other general input on it. But some things might change. Probably nothing general, but perhaps some details.

From: Corey Minyard <minyard@acm.org>
Date: Tue, 13 Nov 2001 20:53:23 GMT
Subject: Ada Structure Library 1.4 release
Newsgroups: comp.lang.ada

I have just put a new version of AdaSL on SourceForge (http://adasl.sf.net). This add the following:

* A reference counting pointer
* A rework of the string tokenizer to make it more usable.
* A calendar package

The biggie here is the calendar package. It does pretty much anything you want with a Gregorian calendar, including leap seconds. If you do fancy processing across timezones or back in time, this is the package for you. I'd appreciate any commentary on this, like ease of use, understandability, etc.

Also, it generates a timezone file from the zone info files supplied with glibc. [...] It contains all the timezones you could possibly imagine back to when timezones started. There is a much smaller simplified version that only contains the current timezone data (no historical information). [...]

Data Structure Packages

From: Jack Beidler <beidler@cs.scranton.edu>
Date: Mon, 15 Oct 2001 17:54:58 -0400
Organization: University of Scranton
Subject: Re: Ada Question
To: team-ada@acm.org

> Just a quick question about Ada95. Is there such a thing as a Collection Object, or an equivalent data structure [...]. Either included in Ada, or coded and available somewhere as a library. [...]

There are several sources of data structure packages in Ada, you can find the Booch components, or the packages I have used for years at http://www.cs.uofs.edu/~beidler/Ada/index.html

John (Jack) Beidler, Ph.D., Professor of Computer Science, Computing Sciences Department, University of Scranton, Scranton, PA 18510, Voice: (570)941-7774, Fax: (570)941-4250

From: aebrain@austarmetro.com.au
Date: Tue, 16 Oct 2001 09:53:46 +1000
Subject: Re: Ada Question
To: team-ada@acm.org

Included in Ada? No. Just as the Java Classes for such are not part of the Java Language, nor C++'s STL part of C++.

Coded and available somewhere in a library? Yes. The difficulty is in finding exactly what type you want. [...] See http://www.adapower.com/links.html for just some of the many libraries of free Ada software.

[Choosing is] made more difficult because of Ada's power to control things. Most collection objects in most languages don't provide any facilities for saying what the maximum size of the collection is, how space is to be managed, garbage collected etc. It's left entirely up to each implementation.

There are such "simple user-friendly" packages available for Ada too. Implementation of a simple stack, list etc is as trivial as "Hello World", any basic textbook will contain one. But there are many others, some of which are for polymorphic types, others with controlled space allocation, automatic space reclamation, reference counting etc. For a simple Generic way of doing things, try the GWU Generic_List by Feldman http://www.cs.uofs.edu/~beidler/cmps144/feldman/listgene.html

Ehud Lamm's http://www.adapower.com/alg/poly_list.zip gives the whole code, specs and bodies for one such which might be of use to you if you don't want to use generics for some reason.

Of course there's also the famous Booch Components available at http://www.adapower.com/booch/ which has amongst other things: Bounded, Dynamic and Unbounded Bags, Collections, Ordered Collections, Deques, Maps, Queues, Ordered Queues, Rings, Sets, Stacks; Directed and Undirected Graphs; Single and Double Lists; AVL Trees; Binary Trees; etc etc etc

10 minutes searching should find you a plethora of others. I think one of these should do :-)

From: Sergei Lodyagin <ladyagin@acm.org>
Date: Tue, 16 Oct 2001 09:42:07 +0300
Subject: Re: Ada Question
To: team-ada@acm.org

There is the SGL library for Ada (STL for C++ is an analog of it).

[See "Standard Generic Library (SGL)" in AUJ 20.1 (April 1999), p.15. -- dc] I use it in my projects and have several patches to the original version (you can find the original in the internet). The patches fix some serious bugs and inconviniences. If somebody is interested in it I can put the patched version on some internet site.

Sergei Lodyagin, Software Developer.

From: "Marc A. Criley" <marccriley@earthlink.net>
Date: Tue, 16 Oct 2001 06:24:50 -0400
Subject: Re: Ada Question
To: team-ada@acm.org

And let me just add Corey Minyard's "Ada Structured Library" (http://adasl.sourceforge.net). Complete and well-documented, besides my own stuff it's been used on at least one Air Force system that I worked on at LockMart.

Marc A. Criley, Senior Staff Engineer, Quadrus Corporation, www.quadruscorp.com

PragmARC - PragmAda Reusable Components

From: Jeffrey Carter <jrcarter@acm.org>
Date: Sat, 03 Nov 2001 23:43:57 GMT
Subject: New Release of the PragmAda Reusable Components
Newsgroups: comp.lang.ada

A new release of the PragmARCs is now available at http://home.earthlink.net/~jrcarter010/pragmarc.htm

[See also AUJ 22.2 (June 2001), p.70. -- dc]

This release provides greater client control over the priorities used by protected structures and task decouplers, as well as introducing a couple of new components. See the file readme.txt for more information.

Jeffrey R. Carter, PragmAda Software Engineering

From: Jeffrey Carter <jeffrey.carter@boeing.com>
Date: Mon, 12 Nov 2001 17:37:30 GMT
Organization: The Boeing Company
Subject: Re: looking for a fast re-usable non-pointer data structure ....
Newsgroups: comp.lang.ada

> Does anyone know of reasonably fast data structure to store a large amount of records.

It would help if you provided more information about what precisely you want. PragmARC.Queue_Bounded is fast, non-pointer, and a data structure, but since you're using an AVL tree I doubt that it does what you want.

Balanced tree structures tend to be slow for insertion and deletion, but fast [O(log N)] for searching. You might be interested in a skip list. Skip lists are approximately O(log N) for searching, and O(1) for insertion and deletion; I do not understand why people continue to use balanced trees given the existence of skip lists. See PragmARC.Skip_List_Unbounded for an implementation of a skip list.

It should be reasonably simple to implement a bounded skip list, just as it is to implement a bounded list.

SAL - Stephe's Ada Library

From: LeakyStain <leakstan@erols.com>
Date: Sun, 18 Nov 2001 18:02:23 -0500
Subject: Update of SAL
Newsgroups: comp.lang.ada

I've posted a new version of SAL to my website: http://users.erols.com/leakstan/Stephe/Ada/sal.html

[See also "Dynamically Growing Arrays" in AUJ 22.2 (June 2001), p.70. -- dc]

There are a couple new packages, and the sorted tree now supports three options for duplicate keys; error, allow, ignore.

Ada compilers for VMS

From: Simon Clubley <simon_clubley@excite.com>
Date: Mon, 10 Sep 2001 12:01:31 GMT
Subject: Re: Where is GNAT for VMS?
Newsgroups: comp.lang.ada

[In response to a request for the GNAT port for a "VAXstation" running VMS: -- dc]

> It looks to me like it is at: ftp://ftp.cs.nyu.edu/pub/gnat/private/old/openvms/

[...], the port at the above address is for [OpenVMS] Alpha only and no port for VAX exists. In case the original poster knows this and was using "VAXstation" as a generic term, the following may be helpful:

Before installing the above kit, you need to install the Ada Predefined Libraries for GNAT as well as the current C RTL patch kit. You can find the Predefined Libraries at: http://www.openvms.compaq.com/commercial/ada/ and follow the link under "Ada 95 Utilities". [...] You can find the C RTL patch kit for your version of VMS by starting at: ftp://ftp.service.digital.com/public/vms/axp/

The debugger in use is GDB, but GDB is not supplied as part of the public kit. [...] If you do find a GDB for VMS, please post it's location here, as I would be interested in finding it.

From: Tucker Taft <stt@avercom.net>
Date: Mon, 17 Sep 2001 09:53:36 -0400
Organization: AverStar (formerly Intermetrics) Burlington, MA USA
Subject: Re: Where is GNAT for VMS?
Newsgroups: comp.lang.ada

> Does GNAT work at all on a VAX machine? If not which Ada compiler system is usable?

Larry Kilgallen is in the process of trying to rehost our AdaMagic compiler to Vax/OpenVMS. I believe he has "Hello World" working. You should contact him for more details.

Tucker Taft, stt@avercom.net, http://www.avercom.net, Chief Technology Officer, AverCom Corporation (A Titan Company), Bedford, MA, USA (AverCom was formerly the Commercial Division of AverStar: http://www.averstar.com/~stt)

From: Kilgallen@SpamCop.net (Larry Kilgallen)
Date: 17 Sep 2001 11:59:32 -0500
Organization: LJK Software
Subject: Re: Where is GNAT for VMS?
Newsgroups: comp.lang.ada

> Can I get DECAda somewhere?

You can still purchase Compaq Ada (83) license for VAX or Alpha. If your purpose qualifies for the VMS hobbyist program the cost is zero.

From: "DuckE" <steved94@home.com>
Date: Fri, 21 Sep 2001 00:36:34 GMT
Subject: Re: Where is GNAT for VMS?
Newsgroups: comp.lang.ada

> If not which Ada compiler system is usable?

Check http://www.irvine.com/native.html It appears they have a VAX/VMS targeted Ada 95 Compiler.

[In another message, Tucker Taft wrote: -- dc]

This compiler passed ACVC suite 2.0.1, which included Ada 95 tests. The earlier ACVC suite 2.0 allowed compilers to pass only Ada 83 tests, but that policy was dropped for 2.0.1.

GNAT 3.13p Binaries for OS/2

From: dwparsons@t-online.de (Dave Parsons)
Date: Sun, 30 Sep 2001 15:47:41 +0200
Subject: GNAT 3.13p binaries for OS/2 available
Newsgroups: comp.os.os2.programmer.misc, comp.lang.ada

For any of you who may not have noticed, GNAT 3.13p binaries for OS/2 are now available at ftp://cs.nyu.edu.

The pub/gnat/3.13p/README.OS2 reads as follows: There is a contributed port for GNAT 3.13p for OS/2 in contrib/os2

The pub/gnat/3.13/contrib/os2/README.txt reads as follows:

This port of GNAT 3.13p for OS/2 has been contributed by: David William Parsons, dwparsons@t-online.de

The documentation has been reformated for OS/2 INF format by: Christian Hennecke, christian.hennecke@os2voice.org

See gnat-3.13p-os2-bin-20010916.txt and gnat-3.13p-os-docs.txt for more information. The files in this directory essentially mirrors what is at ftp://unixos2.org/pub/unix/devtools/emx+gcc/gnat for GNAT 3.13p

In addition ftp://unixos2.org/pub/unix/devtools/emx+gcc/v0.9d contains some EMX elements that may be required for this port of GNAT.

The file readme.313p-os2 in the compiler archive provides more information.

Thanks to all concerned at ACT and NYU.

Plan for GNAT 3.14p Version

From: dewar@gnat.com (Robert Dewar)
Date: 22 Sep 2001 17:29:52 -0700

Subject: Re: Will there be a 3.14p version of GNAT?
Newsgroups: comp.lang.ada

> It appears that the non-public version of GNAT 3.14 has been around for a while now. I'm curious, will there be a 3.14p (public) release of GNAT?

The current plan is as follows. We are going to make 3.14p available soon. If we cannot find the time to build the binaries, then at the least we will provide the 3.14p sources so that others can build the binaries.

This will be the last release in that form. After that, the GNAT sources will be part of the GCC 3.x release, and various people will build public releases from these sources, as happens for GNU C today.

Robert Dewar, Ada Core Technologies

From: dewar@gnat.com (Robert Dewar)
Date: 30 Sep 2001 09:00:53 -0700
Subject: Re: Will there be a 3.14p version of GNAT?
Newsgroups: comp.lang.ada

> [...] think that the GNUAda site (http://www.gnuada.org/) would be the natural alternate clearing-house for precompiled GNAT binaries. It might even be most logical place for the sources too...

No, the primary location for the sources will be the gcc site, since the GNAT sources will be part of the FSF distribution of gcc. Of course anyone can mirror these sources anywhere they like, but the primary site will always be the gnu.org site.

GNAT Sources Contributed to GCC CVS Repository

From: Laurent Guerby <guerby@acm.org>
Date: Wed, 3 Oct 2001 19:58:55 +0200
Subject: Re: ALT RPMs in next SuSE distribution.
To: "GNAT Discussion List" <gnatlist@lyris.seas.gwu.edu>

[...] In the series of great news, GNAT sources have been commited yesterday into the GCC CVS repository. [...]

URL: http://www.gnu.org/software/gcc/gcc.html
Subject: GCC Home Page - GNU Project - Free Software Foundation (FSF)

[...] In April 1999, [...] GCC was renamed from the "GNU C Compiler" to the "GNU Compiler Collection" and received a new mission statement.

Currently GCC contains front ends for C, C++, Objective C, Chill, Fortran, and Java as well as libraries for these languages (libstdc++, libgcj,...). The next major release, GCC 3.1, will also include an Ada front end.

We want to work closely with developers to help and encourage them to contribute changes for inclusion in GCC. We thus provide access to our development sources with weekly snapshots and anonymous CVS.

We will provide regular, high quality releases. We want those releases to work well on a variety of native (including GNU/Linux) and cross targets and use an extensive test suite as well as various benchmark suites and automated testers to maintain and improve quality. GCC 3.0.2 is the current release.

News/Announcements [...]

October 2, 2001: Ada Core Technologies, Inc, has contributed its GNAT Ada 95 front end and associated tools. The GNAT compiler fully implements the Ada language as defined by the ISO/IEC 8652 standard. [...]

Ada Compiler Variety

From: Richard Riehle <richard@adaworks.com>
Date: Mon, 08 Oct 2001 13:25:38 -0700
Organization: AdaWorks Software Engineering
Subject: Re: is Ada dying?
Newsgroups: comp.lang.ada

[In reply to a remark that there were more Ada 83 compiler vendors than there are Ada 95 compiler vendors: -- dc]

[...] Someone might read this as an indication that there are fewer compilers for Ada 95 than Ada 83. What has happened is quite different.

Many of the compilers shown were developed in-house by companies who needed a "checkbox" compiler. I have been told by the senior management of a couple of these companies that the only reason for having a validated Ada compiler is so they could respond to an RFP by checking off the box labled, "Validated Ada." Many of these compilers were designed on top of other compilers, leveraging someone else's technology. If one were to carefully examine the source of these in-house compilers, it would soon become clear that only a few compilers were actually in place, and those targeted to a wide number of computers. Often, the compiler was licensed so the hardware manufacturer could label it with their own proprietary name.

What has happened with Ada 95 is a more realistic organization of the compiler industry. Some compiler publishers have consolidated, hardware manufacturers have seen the folly of trying to be experts in Ada compiler development, the pricing structures have changed, and those who were simply unprofitable failed to make the transition to Ada 95.

One other detail needs to be noted. When Ada was a mandated language instead of an optional one for DoD projects, some compiler publishers saw the mandate as an opportunity to charge outrageous licensing fees for their compilers. Also, since they could get these fees from the DoD, they had little incentive to seriously address the commercial market where those kinds of fees were unacceptable. With a few exceptions, these compiler publishers have been forced to adjust their licensing fees to more realistically reflect the choice now available to DoD software developers.

Ada Cross Compilers for 1750 Processor

From: Stephen Leake <stephen.a.leake.1@gsfc.nasa.gov>
Date: Tue, 30 Oct 2001 12:28:50 -0500
Subject: Re: Cross compilers for 1750?
To: team-ada@acm.org

> I've just received an inquiry regarding the availability of cross compilers for 1750 processors. Can anybody give me a quick indication of any such compilers currently in the marketplace, along with the host processor/OS environment it/they require?

I'm currently on a project using Ada on a 1750. The only compiler available until recently is from DDCI (used to be Tartan). It is basically Ada 83, with some Ada 95 features creaping in. It runs on a Solaris host; I'm not sure if they support other hosts.

There is now a port of GNAT, that I have not used at all. See http://www.xgc.com/. It appears this is _not_ supported by ACT, but you should check it out.

[...], I would strongly recommend using a different chip. The 1750 was designed for Ada 83. Unfortunately, that includes not supporting unsigned arithmetic! It also provides only 64k words for data and code. There are many better chips out there, some with similar power requirements.

Hmm, maybe you mean the extended 1750, with a memory map module. That goes to more than 64k words. I'm not sure if the same compiler supports both versions, or which version the GNAT port supports. I'd also recommend against that version. It is _not_ a flat memory space; it requires manual participation in the link process to arrange the memory pages. Error-prone, and not worth it.

It may be that the GNAT port manages to hide the chip's problems, but if you have any choice of chip, try for a different one.

From: "Vlietstra, Joe" <Joe.Vlietstra@aerojet.com>
Date: Tue, 30 Oct 2001 09:38:24 -0800
Subject: Re: Cross compilers for 1750?
To: team-ada@acm.org

XGC (http://www.xgc.com). M1750 open-source Ada compiler is based on GCC/GNAT. Runs on Solaris and Linux.

DDC-I (http://www.ddci.com). Ada 83 compiler (originally Tartan Ada). Runs on Solaris and VAX/VMS.

Joseph P Vlietstra, Northrup Grumman Space Systems, 1100 West Hollyvale Street, Azusa, CA 91702, joevl@aerojet.com, Tel (626) 812-2865, Fax (626) 812-1290

From: "J.P. Kermode" <kermode@captec.ie>
Date: Wed, 31 Oct 2001 13:57:48 -0000
Subject: Re: Cross compilers for 1750?
To: team-ada@acm.org

There is also the TLD compiler. Currently Ada83 but Ada95 based on Gnat is in pipeline. The TLD Ada83 compiler has been (and is being) used for European spacecraft, e.g. ISO, SOHO, Huygens, XMM, Integral, Rosetta, etc.

Primary host is Sun/Solaris - other hosts are supported. I am not sure but the 1750B target could also be supported (includes unsigned arithmetic).

64Kword memory limitation is a problem, but then good quality efficient code can be written. The ISO AOCS SW (including full SW autonomy) fitted in less than this - both code and data; and there was room for a major post-operations SW update!

Contact point is Terry Dunbar at tldworks@bigplanet.com

CAPTEC, Computer Applied Techniques Limited, 3 St. James's Terrace, Malahide, Co. Dublin, Ireland, Tel: +353-1-8450921, Fax: +353-1-8450136, E-mail mail@captec.ie, Web http://www.captec.ie

GNAT for Mac OS X

From: dewar@gnat.com (Robert Dewar)
Date: 31 Oct 2001 02:01:28 -0800
Subject: Re: Is there an Ada 95 compiler for Mac OS X?
Newsgroups: comp.lang.ada

> I was wondering if anyone out there knows of an Ada 95 compiler for use on Mac OS X (the new Unix MacOS). Is a GNAT port planned, or does anyone know if Green Hills, OCS et. al. are planning a compiler for Mac OS X?

I do not know of any commercial efforts to generate a compiler for Mac OS X. There is indeed a volunteer effort, by Jim Hopper, to generate such a compiler, and ACT is working with him on this effort.

If you are interested in a commercial compiler with support and full tool set etc, I would suggest you contact the various vendors to let them know of your interest.

From: jim <jim_evart@yahoo.com>
Date: Wed, 31 Oct 2001 14:16:57 -0500
Subject: Re: Running x86 Linux GNAT on MacOS under Virtual PC
Newsgroups: comp.lang.ada

> [...], in fact the Mac OS X port is quite tricky (ask Jim Hopper!) and is much more than just a recompilation!

[See also AUJ 22.3 (September 2001), p.140, for more information on that port. -- dc]

Easiest way to run GNAT on OS X right now is to get Codebuilder from Tenon which runs fine under classic environment. However I know Mike Feldman uses the Windows version of GNAT under Virtual PC.

Matrix Package

From: Peter Hermann <ica2ph@csv.ica.uni-stuttgart.de>
Date: 18 Sep 2001 13:55:25 GMT
Organization: Comp.Center (RUS), U of Stuttgart, FRG
Subject: Re: matrix multiplication using 2d arrays
Newsgroups: comp.lang.ada

> Can anybody tell me how to multiply 2 matrices together

http://www.csv.ica.uni-stuttgart.de/ftp/pub/ada/ica/format/

Peter Hermann, Pfaffenwaldring 27, D-70569 Stuttgart, Uni Computeranwendungen, Tel +49-711-685-3611, http://www.csv.ica.uni-stuttgart.de/homes/ph/

GNU.Jif - New Ada Graphics Image Library

From: Paul Pukite <puk@umn.edu>
Date: Sun, 23 Sep 2001 11:58:48 -0500
Subject: NEW Ada graphics image library (GNU.Jif)
To: "GNAT Discussion List" <gnatlist@lyris.seas.gwu.edu>

This message is to announce the availability of a new package implemented for the GNU/Ada library hierarchy: GNU.Jif

Jif is a graphics interchange format library which enables one to generate GIF (pronounced jif) data streams on the fly. The main image type derives from controlled and has dispatchable semantics so that generated data strings can be rerouted to derived Ada apps.

Go to the top of this URL to see what it can do and to download the source: http://umn.edu/~puk

This package should get quite a buzz at the SigAda conference to be held in Minneapolis later this week. In my opinion it is an excellent fit within an XML environment. Try to catch me if you plan to attend.

AdaGraph Revised Version

From: "Kester, Rush W." <Rush.Kester@jhuapl.edu>
Date: Mon, 5 Nov 2001 12:44:53 -0500
Subject: RE: Intel-OA: Basic Graphics Representation on OA
To: intel-objectada@sf.aonix.com

[See also "AdaGraph v0.6 - High-Resolution Color Graphics" in AUJ 22.2 (June 2001), p.72. -- dc]

The documentation for AdaGraph see http://home.trouwweb.nl/Jerry/adagraph.html

There is a version for ObjectAda 7.1 at either http://home.trouwweb.nl/Jerry/ag05oa71.zip ftp://ftp.seas.gwu.edu/pub/ada/windows95/ag05oa71.zip

Martin Carlisle has a revised version see http://www.usafa.af.mil/dfcs/bios/mcc_html/ada_stuff.html ftp://ftp.usafa.af.mil/pub/dfcs/carlisle/adagraph2000/adagraph2000-install.exe

[Revised version allows drawing off the edge of the screen, moves the origin to lower left corner, etc. -- dc]

Rush Kester, Software Systems Engineer, AdaSoft at Johns Hopkins Applied Physics Lab., phone: (240) 228-3030, fax: (240) 228-6779, http://hometown.aol.com/rwkester/myhomepage/index.html

Finder - Ada Web Crawler

From: tmoran@acm.org
Date: Thu, 13 Sep 2001 03:58:51 GMT
Subject: Ada web crawler
Newsgroups: comp.lang.ada

David Botton has kindly posted finder.zip at www.adapower.com/os/finder.html.

It's source plus (Windows) executable for a program that crawls a site checking links. Thus "finder www.adapower.com" will scan the adapower site, following links to local html files and noting links to other files.
"finder www.adapower.com/os" will scan just the "os" directory, treating any links outside that as "foreign", to be noted, but not scanned.

Speed is of course highly dependent on internet access speed. The program is not polished, and still contains some capabilities that were needed for a specific application, but the source code is there for your customization.

AdaDoc - Html Generator for Ada Package Specifications

From: "toa$t" <rogspr@newdeal.ch>
Date: Sun, 21 Oct 2001 15:38:29 +0200
Subject: AdaDoc (make a html file from a package specification for documentation purposes)
Newsgroups: comp.lang.ada

AdaDoc is a tool for the Ada 95 developer. AdaDoc makes an html file from a package specification for documentation purpose.

For Linux & Win32: http://sourceforge.net/projects/adadoc/

[From a later posting: -- dc]

> Very nice. But is there a way to get Adadoc to hyperlink different packages in a project? As I understand it can create individual html files for each .ads. I'm thinking more in the way gnathtml does it? [Gnathtml is included in GNAT distributions. -- dc]

It's not possible to hyperlink different packages (only if you select it). gnathml is not for the same use. AdaDoc use is to present one specific package (promotion).

AWS 1.1 - Ada Web Server Component

From: Pascal Obry <p.obry@wanadoo.fr>
Date: 29 Oct 2001 18:12:15 +0100
Subject: ANNOUNCE : AWS 1.1
Newsgroups:
comp.lang.ada,fr.comp.lang.ada

AWS - Ada Web Server, 1.1 release
Authors: Dmitriy Anisimkov, Pascal Obry

Dmitriy Anisimkov and I are very happy to announce the availability of the AWS 1.1 release. The API could change slightly at this stage but should be fairly stable now.

AWS stand for Ada Web Server. It is not a real Web Server like Apache. It is a small yet powerful HTTP component to be embedded in any applications. It means that you can communicate with your application using a standard Web browser and this without the need for a Web Server. AWS is fully developed in Ada with GNAT.

[See also "AWS 1.0 - Ada Web Server Component" in AUJ 22.3 (September 2001), pp.142-143. -- dc]

Here are the main changes:

- Server push implementation. Tested only with Netscape Navigator.
- SOAP - beta implementation of SOAP. Support all SOAP types. This implementation has been validated through http://validator.soapware.org/ and therefore should be quite inter-operable with other SOAP implementation. This implementation covers the SOAP client interface and as all supports to build SOAP servers. Versions that validate on http://validator.soapware.org/ are the AWS version string (AWS.Version) catenated with the SOAP version string (SOAP.Version).
- Add accept queue size parameter to help building heavy loaded servers.
- Fix the Runme NT service demo.
- Web Servers is started only if needed (during Server.Start) call and not when declaring the HTTP objects.
- Max Connection is not anymore a discriminant. This parameter is set with the Start routine. This change is not upward-compatible, but it is worth it since now, it is possible to change the server configuration dynamically. What need to be changed: (1) remove the discriminant on each HTTP objects; (2) pass the number of maximum connections (was the discriminant) in the Server.Start call. This will make the server configured the very same way.
- Handle User_Agent and Referer HTTP headers.
- Add message size in the log files (last field). Now the log format is 100% compatible with the standard ones (Apache and Internet Information Server).
- Add server start time in the status page.
- Add support for user's log.
- Properly terminate task Session.Clean up and release associated memory. Fix a memory leak.
- Properly wait for tasks termination before releasing memory. Fix memory leak.
- Improves the documentation.
- Install AWS as a library (libaws.a)
- As always some minor bugs have been fixed but are not listed here. See src/ChangeLog and SOAP/ChangeLog.
- Change the build procedure, should be easier and it is cleaner. See documentation.
- First version of the regression tests suite. This will help keeping AWS more stable.
- Add timeouts support for the AWS client interface. Because of this the AWS.Client.Create routine has its spec changed (it was a function it is now a procedure).
- In AWS.Client, default retry count is set to 0 (was 1 before). Now the AWS.Client routines wont try more than once to get the data by default.
- Properly handle textual (text/html, text/xml...) data that is chunked encoded.
- Fix SSL support in AWS. The SSL layer should now be as reliable as the standard socket one.
- Update distribued Win32 OpenSSL library to version 0.9.6b. Also now there are built as DLL.

[...]

Pointers [updated]:

- AWS User's Mailing List: http://lists.act-europe.fr/mailman/listinfo/aws
- AWS Home Page (sources and documentation): http://libre.act-europe.fr/
- Templates_Parser sources: Templates_Parser module (sources and documentation) is provided with AWS distribution. Latest version of this module and the documentation can be found at: http://perso.wanadoo.fr/pascal.obry/contrib.html http://perso.wanadoo.fr/pascal.obry/templates_parser.html

  Templates_Parser is a very useful add-on for AWS. You should have a look at it if you plan to develop a Web service. Templates_Parser permits to completely separate the HTML design from the Ada code. [...]
- XMLada (optional): You need this library only if you want to use AWS SOAP feature. You need at least XMLada 0.6. http://libre.act-europe.fr/

  XMLAda 0.6 has some memory leaks. This has been fixed now, so with future version of XMLAda it will be possible to build long-lived servers.
- Socket binding: for Win32: http://perso.wanadoo.fr/pascal.obry/contrib.html for UNIX: http://www.rfc1149.net/devel/adasockets
- POSIX Binding (optional) : for Win32: http://perso.wanadoo.fr/pascal.obry/contrib.html for UNIX: http://www.cs.fsu.edu/~baker/florist.html
- OpenSSL library (optional) : Sources for UNIX or Win32: http://www.openssl.org Binaries for Win32 with GNAT 3.13 (and later): included with the main AWS distribution.

  Note that we have used and we distribute (for Win32 platform) OpenSSL version 0.9.6b with this AWS release. OpenSSL have been built with GCC version 2.95.2 with optimization (-O3) on. See OpenSSL license (docs/openssl.license).
- Windows Services API (optional): To build the runme demo as a Windows NT/2000 service you must download the services API made by Ted Dennison for his SETI@Home project. http://www.telepath.com/dennison/Ted/SETI/SETI_Service.html

Reporting bugs: you can report bugs to Dmitriy Anisimkov (anisimkov@yahoo.com) and Pascal Obry (p.obry@wanadoo.fr).

It would be nice if you could also sent us a note if you are using AWS just to know if it is used at all or not :) And if you are ok, we'll add an entry for your project in the next section.

AWS User's Mailing List:

A good way to keep informed of AWS news and to share experience with other AWS users is to register to the AWS dedicated mailing list. See: http://lists.act-europe.fr/mailman/listinfo/aws

AWS uses:

- Internet Currency Trading System at www.actforex.com by Dmitriy Anisimkov

  This is a server used to keep historical data about currency trading to build charts of currency prices. The charts viewer part is written in Java and loaded through AWS. This server can be reach on the Internet. Ongoing work is done to base this development on AWS framework only and to remove all the Java layers. It is also interesting to note that this is a heavy loaded server, it has something like 40 to 50 requests per second.

- [For more projects using AWS, see a.o. "AWS - Ada Web Server Package" in AUJ 22.2 (June 2001), pp.75-77. -- dc]

Thanks to all who have reported bugs and have sent us patches.

Dmitriy & Pascal.

Pascal Obry, Team-Ada Member, 45, rue Gabriel Peri, 78114 Magny Les Hameaux, France, http://perso.wanadoo.fr/pascal.obry

Tools for Creating Ada Bindings to C Headers

From: Hal Hart <Hal.Hart@trw.com>
Date: Fri, 26 Oct 2001 12:26:00 -0700
Subject: Re: C-to-Ada conversion
To: team-ada@acm.org

What do Teamers think is the best tool these days to assist converting C programs to Ada. (Yes, someone here at TRW is in the position of needing to do so. :-) Thanks in advance for any help, -HH

[From another message: -- dc]

PS: Is the old c2ada program from the 80's regarded as "reputable"? If so, where do we get it now? I see no immediately obvious links (to c2ada or the subject in general) from the SIGAda or PowerAda websites...

From: Mark Lundquist <mlundquist2@home.com>
Date: Fri, 26 Oct 2001 13:35:49 -0400
Subject: Re: C-to-Ada conversion
To: team-ada@acm.org

http://www.averstar.com/~stt/bindings/c2ada/c2ada.html
I don't know how "old" this c2ada is (nor how "reputable") :-)

[See also "Former "Intermetrics" Bindings" in AUJ 21.1 (April 2000), pp.17-18, and "Tools for Creating Ada Bindings to C Headers" in AUJ 20.4 (January 2000), pp.235-236. -- dc]

From: David Botton <David@botton.com>
Date: Fri, 26 Oct 2001 16:33:22 -0400
Subject: Re: C-to-Ada conversion
To: team-ada@acm.org

Cbind ported to windows (source in package) http://members.tripod.com/vagul/ I use this as part of automated makes to convert the headers generated by Resource Editors. I don't know how well it works beyond that.

C2ada [ported to linux -- dc] http://home.pacbell.net/nma123/

From: Chad Bremmon <chad.bremmon@parnassussolutions.com>
Date: Wed, 31 Oct 2001 09:54:44 -0500
Subject: Re: C-to-Ada conversion
To: team-ada@acm.org

> [...] I was just curious what's the advantage in converting the C programs to Ada? I mean, if the C program works, isn't it better to just import the C program?

I'm tending to agree with Steven on this one. What I recommend is starting out using Ada pieces to glue the C components that you have. Each compiler has an "interfaces.c" package that you can use to glue everything together. Then piece by piece, you could rethink each component and rebuild it. Divide and conquer supporting the interfaces.

Chad Bremmon, Senior Solution Specialist, Parnassus Solutions

Using Sockets in Ada

From: Preben Randhol <randhol+abuse@pvv.org>
Date: Wed, 17 Oct 2001 10:17:55 +0000 (UTC)
Organization: Norwegian university of science and technology
Subject: Re: sockets in Ada
Newsgroups: comp.lang.ada

> Can someone tell me were can I find documentation about the use of sockets in Ada language? I'd like to build a Server program able to comunicate whith different client programs.

Here you can find AdaSockets. I believe GNAT also has a socket package now, but is not yet in the public version of GNAT. http://www.infres.enst.fr/ANC/

Here is a Simple MUD that uses the Ada Sockets http://members.aol.com/drveg/mud/

Here from the "Big Online Book of Linux Ada Programming", though it looks like it also does some thin binding to the C lib ie. not using AdaSockets. http://www.vaxxine.com/pegasoft/homes/16.html#16.23

From: john.mccabe@emrad.com (John McCabe)
Date: Wed, 17 Oct 2001 11:00:01 GMT
Organization: Emrad Ltd
Subject: Re: sockets in Ada
Newsgroups: comp.lang.ada

For a Winsock 2 binding go to: http://www.adapower.com/reuse/winsock2.html

For BSD Socket bindings (including one for Win32) go to: http://www.adapower.com/os/bsd-sockets.html

There are some examples here (I think) of using sockets. Although the basics of using Sockets on BSD and Win32 is pretty much the same, there are differences which need to be considered if you want to do anything interesting with them (especially Winsock). For instance Winsock uses macros FD_SET, FD_CLR, and FD_ZERO, but the underlying structure is totally different between Winsock and BSD. I would assume that, as long as you use one of the above bindings, these differences should be catered for. You may already know a fair bit about sockets, but if not....

For resources related to Winsock 2, have a look at: http://www.stardust.com/winsock/index.htm

For more information on BSD sockets, try a book like: UNIX Network Programming Vol 1: Networking APIs - Sockets and XTI by W. Richard Stevens. Published by Prentice Hall, ISBN 013490012X

From: "Marc A. Criley" <mcqada@earthlink.net>
Date: Wed, 17 Oct 2001 11:52:34 GMT
Organization: Quadrus Corporation
Subject: Re: sockets in Ada
Newsgroups: comp.lang.ada

You may wish to examine "Ada Web Server" (http://perso.wanadoo.fr/pascal.obry/aws.html) as both a way to understand how to do socket programming in Ada, and as a very useful and capable client/server tool itself.

From: Samuel Tardieu <sam@rfc1149.net>
Date: Mon, 22 Oct 2001 20:17:40 +0200
Subject: Re: sockets in Ada
Newsgroups: comp.lang.ada

The latest AdaSockets package works now on both Unix and Windows platforms, thanks to Dmitriy and Pascal [of AWS fame -- dc]. It can be fetched from: http://www.rfc1149.net/devel/adasockets/

From: whraven@usenet-access.com (Richard Pinkall-Pollei)
Date: 17 Oct 2001 11:38:47 -0500
Subject: Re: sockets in Ada
Newsgroups: comp.lang.ada

The Florist package from Florida State University implements POSIX standard sockets for Ada. However, unless you've won the lottery, or have a boss who's

willing to fork over cash for documentation (POSIX standards documents are expensive), you'll need to study the source files to understand how to use it, and even then, there'll be stuff you just have to discover for yourself by trying it.

Others have already mentioned the AdaSockets package.

The third option is to use `Interfaces.C' to use your OS's native sockets library.

From: tmoran@acm.org
Date: Wed, 17 Oct 2001 17:31:52 GMT
Subject: Re: sockets in Ada
Newsgroups: comp.lang.ada

For a short and simple web server (that uses Claw.Sockets) see: www.adapower.com/reuse/clawweb.html

To see Claw.Sockets itself, download the source from www.rrsoftware.com (The $>0 version has additional stuff, but isn't needed for a simple web server.) Full disclosure: I wrote it so I'm probably biased.

Indexed_IO Package

From: Wesley_Groleau@raytheon.com
Date: Fri, 28 Sep 2001 09:02:30 -0500
Subject: Indexed_IO
To: team-ada@acm.org

Ages ago, I had to port an app from VAX Ada to Verdix. It used a VMS-specific "Indexed_IO" package which had to have a Verdix version created. Due to the usual employee innovation agreements, I can't offer it to the public, but something similar might be a useful thing if someone wanted to re-invent it.

Just imagine some form of search tree and/or hash mechanism, only replace the access types with Direct_IO.Count

From: Dirk Craeynest <Dirk.Craeynest@cs.kuleuven.ac.be>
Date: Fri, 28 Sep 2001 21:40:07 +0200
Subject: Re: Indexed_IO
To: team-ada@acm.org

[...] You might be interested in "Indexed sequential files in Ada" at http://www.cs.kuleuven.ac.be/~dirk/ada-belgium/software/#IND on the "Free Ada Software provided by Belgian Ada users" part of the Ada-Belgium web-server.

The text of that item on our software page is:

Indexed sequential files in Ada: a didactical example

Author: Marc A. Gobin, Royal Military Academy, Brussels Status: Source code available. Platforms: portable, tested with Meridian (Ada 83) and GNAT (Ada 95) on PC/DOS. Entry added: 1996/10/23. Entry last changed: 1997/03/16.

Reference: "Indexed Sequential Files in Ada: a Didactical Example", by Marc A. Gobin, Ada-Belgium Newsletter, Nov 1996, Pages 85-91, Volume 4

Abstract: As an introduction to a course on implementing data bases, the working of an indexed sequential file system is explained. Indexed sequential files are not included in the Ada reference manual, but can (easily?) be implemented. To serve its purpose the implementation should be easy to explain, easy to use and as efficient as possible. In a paper presented at the 1996 Ada-Belgium Seminar the main features of an indexed sequential package are explained and the different choices and restrictions are justified. The result is a quite efficient package for defining and using the traditional index sequential concepts. Note: the entire package is available as freeware and can be obtained in source form here. The package is Ada83 compatible.

The following files are available:

* indexed.doc: a copy of the author's paper in the Ada-Belgium Newsletter (MS Word 6.0 document);

* indexed.ppt: a copy of the author's presentation at the 1996 Ada-Belgium Seminar (Powerpoint file);

* indexed.ada: the source code.

Check out the URL mentioned above if you would like to download one of these files or to contact the author.

Dirk Craeynest, Offis - Aubay Group, Weiveldlaan 41/32, B-1930 Zaventem, Belgium, Dirk.Craeynest@cs.kuleuven.ac.be (Ada-Belgium), Dirk.Craeynest@offis.be (work), Phone +32(2)725.40.25, Fax +32(2)725.40.12

From: Pascal Obry <p.obry@wanadoo.fr>
Date: Fri, 28 Sep 2001 22:38:37 +0200
Subject: Re: Indexed_IO
To: team-ada@acm.org

> Just imagine some form of search tree and/or hash mechanism, only replace the access types with Direct_IO.Count

Don't imagine, just grab Adbm from my homepage. This is an indexed file implementation using a hash code at the first level and then a B-Tree on disk... don't remember all the details since I have done it long time ago, I have used it a lot and it seems quite stable...

[From another message: -- dc]

As a side note, Adbm is completly written in Ada and is not an interface to gdbm. I have never compared the speed of Adbm and gdbm, yet I think Adbm is quite fast certainly slower than gdbm which has been optimized a lot. Adbm is one component I used a lot in a search engine which is part of numerous projects I have built at work... so I'm confident that it should be working fine.

From: Wesley_Groleau@raytheon.com
Date: Fri, 28 Sep 2001 16:21:12 -0500
Subject: Re: Indexed_IO
To: team-ada@acm.org

So two someones (at least) already DID invent it! :-)

Ada Binding to MySQL

From: Preben Randhol <randhol+abuse@pvv.org>
Date: Sun, 7 Oct 2001 19:21:04 +0000 (UTC)
Organization: Norwegian university of science and technology
Subject: Re: Ada MySQL binding
Newsgroups: comp.lang.ada

> I'm trying to find a free binding to MySQL. [...]

http://gnade.sourceforge.net/

[See also "GNADE - GNU Ada Database Environment" in AUJ 22.3 (September 2001), p.144. -- dc]

From: byhoe@greenlime.com (Adrian Hoe)
Date: 7 Oct 2001 18:17:17 -0700
Subject: Re: Ada MySQL binding
Newsgroups: comp.lang.ada

You can try this:
ftp://ftp.greenlime.com/pub/Ada/

[The file ada-mysql.tar.gz -- dc]

Free Online Game Project in Ada

From: "Christophe" <christophe.dubach@epfl.ch>
Date: Tue, 9 Oct 2001 17:15:55 +0200
Subject: Free Online Game project in Ada
Newsgroups: comp.lang.ada

I've an idea of a online game project in Ada. The game would be a strategic game (like mankind), I think about creating a game under the GPLlicense. The game could use MySQL as Database and OpenGL for graphic (library exists in Ada).

What do you think about that? It could be very good to do such a thing, first of all to build something on Ada and also to do it under GPL license. For the moment it's only an idea but if other people think this could be fun to do it, why not :-) Waiting for your post...

[From another message: -- dc]

The idea would be to create this on Linux, but it will be better to make an OS-Independant game... I know that Glade (which can access to MySQL) and Glut (OpenGL) can run on windows...

I'll create a website to explain more in details what the project would be, and also let people telling their opinion...

[And somewhat later: -- dc]

Check this url :
http://www.freename.f2s.com/adagame.html

I've created this page in a few minutes, so don't expect to have something nice ;-)

From: Pascal Obry <p.obry@wanadoo.fr>
Date: 09 Oct 2001 20:48:42 +0200

Subject: Re: Free Online Game project in Ada
Newsgroups: comp.lang.ada

> The game could use MySQL as Database and OpenGL for graphic (library exists in Ada).

I have some demos and a working binding on my homepage. Look for OpenGL, GLUT and GLAUX keywords on this page: http://perso.wanadoo.fr/pascal.obry/contrib.html

[See also "OpenGL Bindings and Demos" in AUJ 22.2 (June 2001), pp.72-73. -- dc]

From: Lorenzo Micheletto <lorenzomicheletto@libero.it>
Date: Thu, 11 Oct 2001 17:41:11 GMT
Subject: Re: Free Online Game project in Ada
Newsgroups: comp.lang.ada

> The idea would be to create this on Linux, but it will be better to make an OS-Independant game... I know that Glade (which can access to MySQL) and Glut (OpenGL) can run on windows...

For the low_level_graphics/audio/game_input check SDL too (www.libSDL.org). It covers most of the basic needs and has lots of add-on modules. It is written in C (with some add-ons written in C++) but there is an Ada binding. SDL supports accelerated 2D and OpenGL, plus joystick/gamepad/mouse/keyboard IO routines, audio mixer for sound effects,"portable" timing and threads and "raw" cd-rom support (to play audio tracks directly from cdrom) plus other useful things.

[See also "AdaSDL - Binding to Simple DirectMedia Layer (SDL)" in AUJ 22.3 (September 2001), p.141. -- dc]

It runs on Windows, Linux, OS/2, Beos and there is a Playstation II port in progress.

[Preben Randhol <randhol@pvv.org> wrote: -- dc]

Yes. Looks like more and more projects use SDL.

Mine Detector Game

From: Jeffrey Carter <jrcarter@acm.org>
Date: Sun, 04 Nov 2001 22:24:13 GMT
Subject: Mine Detector Game
Newsgroups: comp.lang.ada

PragmAda Software Engineering has released a game called Mine Detector. Written entirely in Ada, it is released under the GNU Public License. Win32 and Linux/x86 executables are available. Full source code is also available, of course.

http://home.earthlink.net/~jrcarter010/mindet.html

Jeffrey R. Carter, PragmAda Software Engineering

From: Tucker Taft <stt@avercom.net>
Date: Tue, 06 Nov 2001 10:34:09 -0500
Organization: AverCom Corp, a Titan company
Subject: Re: Mine Detector Game
Newsgroups: comp.lang.ada

Great job. This is just the kind of thing to help broaden the appeal of Ada.

"Fun with Ada" Lab at AdaPower

From: "Kester, Rush W." <Rush.Kester@jhuapl.edu>
Date: Wed, 14 Nov 2001 12:32:58 -0500
Subject: Re: Mine Detector Game
To: team-ada@acm.org

[...] To all interested in Ada Games:

How about joining the "Fun with Ada" lab at AdaPower? I setup a lab area on David Botton's AdaPower site for getting folks together who want to help promote Ada by showing how much fun you can have with Ada applications. Ada games are a perfect match.

See, http://www.adapower.com/lab/adafun.html for how you can help from anywhere in cyberspace or just track what's going on. To discuss this further please use http://www.adapower.com/lab/adafun/chat.html

My goal is to reach the budding programmers in high school and expose them to software engineering with Ada before they learn too many bad habits. I plan to use various applications that will capture the interest of young programmers (or those young in spirit :-).

For example, controlling model trains, slot cars, and robots. Barry Fagin has developed an interface to allow Ada programs to control the LEGO Mindstorm's robots. see http://www.usafa.af.mil/dfcs/adamindstorms.htm We had a demo of an Ada controlled LEGO Mindstorms robot at the SIGAda 2000 conference hosted here at Johns Hopkins University Applied Physic Laboratory.

Another example is David Wheeler's skeleton of an object-oriented adventure game called "Small" in Ada, http://www.adahome.com/Tutorials/Lovelace/small.htm.

Rush Kester, President Baltimore SIGAda

Distributed Systems Annex and Transport Security

From: David Brown <davidb-cla@davidb.org>
Date: Fri, 28 Sep 2001 16:30:01 GMT
Subject: Glade using SSL.
Newsgroups: comp.lang.ada

I am considering using DSA (with Glade) to implement remote backups and such in my Adump backup software <http://www.davidb.org/adump/>.

[See also "Adump 1.0 - Backup Software" in AUJ 22.3 (September 2001), p.140. -- dc]

I was wondering if anyone has looked in how to implement transport security (such as SSL) into Glade.

I looked at the filter mechanism example for zip, but it appears that the Glade filters assume every chunk of input data makes another chunk of output data. This doesn't apply to SSL, at the beginning an exchange must happen.

The other problem is that Garlic opens multiple sockets between the programs (boot server and such). The port numbers are also not fixed. This would be difficult to use for an administrator that needs to open a port on a firewall.

The other idea I've thought of is to use Garlic as a base and implement a more primitive PCS that only implements the features that I want and uses SSL.

From: Pascal Obry <p.obry@wanadoo.fr>
Date: 28 Sep 2001 20:37:04 +0200
Subject: Re: Glade using SSL.
Newsgroups: comp.lang.ada

> I was wondering if anyone has looked in how to implement transport security (such as SSL) into Glade.

Of course :) Not SSL but DES, RSA and IDEA.

http://lglwww.epfl.ch/Ada/filters/home_page.html

Another solution could be to use AWS and do the transport through HTTPS (SSL 3).

BUSH 0.8 - AdaScript Shell

From: PegaSoft Canada - ALT Drop Box <adalin-l@tiamet.vaxxine.com>
Date: Sun, 30 Sep 2001 16:01:03 -0400 (EDT)
Subject: BUSH 0.8 beta release
To: "GNAT Discussion List" <gnatlist@lyris.seas.gwu.edu>

In the next day I will be posting the source code for my AdaScript shell BUSH. This should be considered a release candidate and I'm looking for any obvious bugs that need fixing prior to the official release of version 0.8. BUSH is available at http://www.vaxxine.com/pegasoft.

BUSH is a Linux/UNIX shell which uses a subset of Ada 95. Version 0.8 is a major upgrade over 0.1, and many sections of the source code have been completely rewritten. [See also AUJ 22.1 (March 2001), p.11. -- dc]

New features are too numerous to list, but include:

* works as a login shell

* standard Text_IO, numeric and string functions implemented

* while, case, loop, enumerated for loop implemented

* TCP/IP sockets

* much greater reliability and performance

* better documentation and improved help command

* no longer a front-end to BASH, now a true shell

* sample shell script: Eliza, the famous AI program

[...]

Ken O. Burtch, http://www.vaxxine.com/pegasoft, ken@tiamet.vaxxine.com, Pegasoft, R.R.#1, Jordan Station, ON, Canada L0R 1S0

From: PegaSoft Canada - ALT Drop Box <adalin-l@tiamet.vaxxine.com>
Date: Sat, 6 Oct 2001 08:52:33 -0400 (EDT)
Subject: BUSH 0.8 beta 2 available
To: "GNAT Discussion List" <gnatlist@lyris.seas.gwu.edu>

A new beta version of my BUSH AdaScript shell is available on the PegaSoft web site at http://www.vaxxine.com/pegasoft/bush-down.html.

* cd command has been fixed

* --restricted (-r) restricted shell option

* several other small bugs tested and fixed

From: Ken Burtch <kburtch@sympatico.ca>
Date: Fri, 19 Oct 2001 12:13:16 -0400
Subject: BUSH 0.8 - Open Source AdaScript Shell released
Newsgroups: comp.lang.ada

BUSH (AdaScript Business Shell) 0.8 Released

BUSH binaries and GPL source code: http://www.vaxxine.com/pegasoft/bush-down.html. BUSH currently runs on HP-UX and Linux. Ports to other platforms are welcome!

New to this Release:

* BUSH tutorial

* Built-in string and numeric packages

* TCP/IP sockets

* Faster performance

See the BUSH Guide for complete details.

Post bug reports to pegasoft@tiamet.vaxxine.com or on the BUSH forum. Please include source code to duplicate the problem. [...]

ELSE - Emacs Language Sensitive Editing

From: Stephen Leake <stephen.a.leake.1@gsfc.nasa.gov>
Date: 29 Oct 2001 12:48:35 -0500
Organization: NASA Goddard Space Flight Center
Subject: Emacs Language Sensitive Editing
Newsgroups: comp.lang.ada

For those who might have missed it, here is the announcement of a version release for Else, which has excellent support for Ada. It was posted on comp.emacs.sources.

Emacs Language Sensitive Editing (ELSE). Access to the package can be obtained from http://www.zipworld.com.au/~peterm

Keywords: template, skeleton, abbreviation

ELSE is a minor mode for Emacs that provides code templates/skeletons/ abbreviations for whatever language major mode is in force in the current buffer (assumes that you have a set of ELSE template definitions for that language, of course! :-)). Very similar in concept to the template, skeleton and others but more powerful and (hopefully) easier to understand for people who are not Elisp experts because: (a) ELSE template definitions are written in an ASCII format rather than Elisp style (with all those "gotchas" of missing closing braces etc :-)); and (b) comes with extensive documentation (40+ page manual).

ELSE comes with language templates for C, Python, Ada (83 and 95), Emacs Lisp, C++ and Java. These template files are in varying degrees of "completeness" and "usability" so choosing just one to trial ELSE may not necessarily be fair to the mode :-). For instance, the C++ and Java templates are in a very primitive state. I don't program in either of these languages so they are pretty much first passes from a program I have written that generates ELSE templates from EBNF. I am only publishing these language templates in the hopes that they can give someone a starting point for further development :-) I know of one developer who has started to use the C++ templates and his changes are available on the web site (in C++-cust.lse).

I consider the Python and Ada templates the most complete sets. The C templates are from a very early era when I first started coding with ELSE and thus reflect a fairly primitive set of templates. I am back into a C environment now but in a maintenance role, so they don't get much of a chance to get a work out :-).

This version has been a very long time in gestation. Users of ELSE will appreciates some of the small, but significent changes that have been made to the package.

As always, if there are any questions, comments, suggestions or requests for help, feel free to contact me at peter.milliken@gtech.com :-) I am more than happy to help people understand and use the package.

Ada Kalinda Operating System

From: "Vincent Morin" <vincent.morin@univ-brest.fr>
Date: Fri, 2 Nov 2001 16:52:46 +0100
Organization: Universite de Bretagne Occidentale
Subject: Ada Kalinda operating system software initial release
Newsgroups: comp.lang.ada

I released an initial version of the Ada translation for the Kalinda OS formerly written in metrowerks Pascal, it is available at http://sourceforge.net/projects/sx-ada-kalinda. Sources can be Gnat or Aonix compiled but system is not operational due to deep transformations in the file/resource system. If anybody is interested in the project or has comment about the sources, all constructive participations are welcome.

I also hope it could give some ideas for AdaOS (I think it is not the direction taken, but at least, I have sources and the old Pascal system was working). Sources are in french (sorry, but as I understand english, I think some english programmers can understand french. I is not great literature!).

Vincent Morin, Laboratoire de Biostatistiques et Informatique Médicale, 22 Avenue Camille Desmoulins, 29285 Brest cedex, France

Auto_Text_IO ASIS Application

From: LeakyStain <leakstan@erols.com>
Date: Sun, 18 Nov 2001 18:00:48 -0500
Subject: Auto_Text_IO ASIS application
Newsgroups: comp.lang.ada

I've posted an ASIS application to my web site: http://users.erols.com/leakstan/Stephe/Ada/auto_text_io.html

It generates Text_IO routines Put and Get, using named notation, when given an Ada package containing types. This makes it much less tedious to write readable unit tests, and provides persistent storage in human readable format.

The tool is released under the GPL, the run-time components under the GMGPL.

There's a switch to generate Ada 83 compatible code; no Get then, because Ada 83 doesn't have Text_IO.Look_Ahead.

The Get routines are not as robust as an Ada compiler; the components must be in declaration order. That's sufficient for reading the output of the Put routines, less good for hand-written inputs. It would be interesting to try to combine this with OpenToken to make it more powerful.

Stephen Leake

Ada-related Products

ACT-Europe - XML/Ada 0.6

From: Emmanuel Briot <briot@act-europe.fr>
Date: Fri, 07 Sep 2001 16:27:20 GMT
Subject: [ANNOUNCE] XML/Ada 0.6 released
Newsgroups: comp.lang.ada

We are happy to announce the release of a new version of XML/Ada (0.6). This is a set of Ada packages that can be used to manipulate XML streams. It includes a full XML parser (including for the DTD part), as well as SAX 2.0 and DOM 2.0 compliant interfaces (please see the web page and the documentation for more information on these interfaces). It also includes a Unicode module to manipulate and convert Unicode streams.

It passes all of the applicable tests of the official XML conformance testsuite. This new release includes an optimized parser (rewritten from scratch since the previous release). This fixes the last problems with the official XML conformance testsuite.

[The software is available at http://libre.act-europe.fr/xmlada/. See also "ACT-Europe - XML/Ada 0.5 Suite of Tools" in AUJ 22.2 (June 2001), p.79. -- dc]

ACT is providing full support for this tool set. Let us know at sales@gnat.com or sales@act-europe.fr if you are interested in evaluating this library for commercial use.

From: Ted Dennison <dennison@telepath.com>
Date: Mon, 10 Sep 2001 14:04:49 GMT
Subject: Re: [ANNOUNCE] XML/Ada 0.6 released
Newsgroups: comp.lang.ada

> I think Emmanuel missed a change which might be important for some people: XML/Ada is now released under the GPL with GNAT modifications.

It is indeed great news that there is now an XML solution we can recommend to people without reservations.

T.E.D.,
http://www.telepath.com/dennison/Ted/TED.html

Aonix - ObjectAda 7.2.1 for Windows

From: "Peter Dencker" <dencker@web.de>
Date: Sun, 11 Nov 2001 20:11:59 +0100
Subject: FREE ObjectAda version 7.2.1 now available!
Newsgroups: comp.lang.ada

[...] For Immediate Release

Contact: Greg Gicca, Director of Product Management, adamark@sd.aonix.com; additional product information: (858) 457-2700, info@aonix.com

Aonix Expands Capabilities of Best-Selling ObjectAda for Windows

SAN DIEGO, September 1, 2001 - Aonix, a leading provider of Windows software development environments, announced today that the latest version of its object-oriented development environment, ObjectAda version 7.2.1, is now available for Windows 98, Windows NT, Windows 2000 and for the embedded ETS Real-Time Win32 operating systems.

Aonix, the leading supplier of quality Ada technology for the Windows platform, says that the new release represents a significant upgrade for over 150,000 worldwide software developers who have used ObjectAda and enjoyed its pacesetting compiler, debugger, browser, editing, and project management capabilities.

The ObjectAda Ada95 and Multi-Language development environment has just become more powerful. While the past release added multi-language editing, project control, and a newer MS Visual Studio look and feel, Version 7.2.1 adds more power to the core product. It includes: a more powerful code generator for better program performance; additional debugging capabilities; and an SCCI (Source Code Control Interface) capability.

The latter provides direct access to the MS SCCI from the ObjectAda IDE/GUI. It allows users to make use of any CM system that conforms to this MS standard. The previous release of ObjectAda offered the capability to add integrations with any external tools via its Customize option.

The new SCCI capability adds the following functions for any underlying CM system: List Files; Keep Checked Out; Comment; Select All / UnSelect All; Get Latest Version; Check Out; Check In; Undo Check Out; Add to Source Control; Remove from Source Control; Show History; Show Differences; Source Control Properties; Invoke External Source Control.

"With SCCI as the industry standard CM interface on Windows, Aonix can offer CM integration to almost any Windows-based CM system from ObjectAda. This new feature provides both a powerful and versatile capability for all our Windows hosted products," states Greg Gicca, Product Manager for ObjectAda.

The ObjectAda free Special Edition is now available on the Aonix web site. To get your copy go to: http://www.aonix.com/content/products/objectada/windows.html and select "ObjectAda for Windows Special Edition" in the right hand download section.

About Aonix

Aonix is a leading international software company providing a comprehensive suite of products and services aimed at managing the complexity of today's business-critical application development and information management. The Aonix Critical Development Solutions division provides a comprehensive suite of software development products that support the customer's development process, methods and tools. The result is minimized risk, improved developer productivity and increased software quality. Along with our native product offerings Aonix offers embedded and full Safety Critical development environments. Headquartered in San Diego, Aonix operates sales offices throughout North America and Europe in addition to a network of international distributors. More information about Aonix can be found on its Web site at www.aonix.com.

From: Ada Marketing <adamark@sd.aonix.com>
Date: Wed, 14 Nov 2001 11:47:53 -0500
Subject: Aonix Expands Capabilities of Best-Selling ObjectAda for Windows
To: team-ada@acm.org

For a copy of the latest free ObjectAda for Windows version 7.2.1, see the below page and select the download button on the right hand frame.
http://www.aonix.com/content/products/objectada/windows.html

The product release description is [at] http://www.aonix.com/content/news/pr_09.01.01.html

From: "Peter Dencker" <dencker@web.de>
Date: Thu, 8 Nov 2001 10:48:57 +0100
Subject: Re: ObjectAda 7.1 Special Edition - [...]
Newsgroups: comp.lang.ada

You may ask your local Aonix office for a free copy of the Special Edition [7.2.1] if you don't have the bandwidth to download it directly from http://www.aonix.com/content/products/objectada/windows.html or via ftp [from] ftp://ftp.aonix.com/pub/ada/public/pal/

Peter Dencker, Sales Manager Aonix GmbH (Germany)

From: "Snodgrass, Britt (NM75)" <Britt.Snodgrass@honeywell.com>
Date: Thu, 15 Nov 2001 09:14:50 -0700
Subject: Re: object ada
To: team-ada@acm.org

> I think the 'Save File As' function is disabled in the _demo_ version of the GUI builder.

With the free ObjectAda 7.2.1 Special Edition, you can now save your GUI builder project. However the project is restricted to two windows, four controls per window, and no ActiveX controls.

Aonix - ObjectAda 7.2.1 for SPARC/Solaris

URL: http://www.aonix.com/content/news/pr_10.26.01.html
Date: Thu, 15 Nov 2001 10:57:00 +0100
Subject: Aonix(r) Announces the Next Major Release of ObjectAda(r) for SPARC Solaris Version 7.2.1

Aonix(r) Announces the Next Major Release of ObjectAda(r) for SPARC(r)/Solaris(r) Version 7.2.1

BOULDER, Colorado, October 26, 2001 - Aonix(r), a member of the Gores Technology Group and a leading provider of Ada 95 software development environments, announced today that the latest version of its object-oriented development environment, ObjectAda(r) version 7.2.1, is now available for the SPARC(r)/Solaris(r) operating system, versions 2.6, 7, and 8.

Aonix, the leading supplier of quality Ada technology, says that this new release represents a significant upgrade for over 150,000 worldwide software developers who have used ObjectAda and enjoyed its pacesetting compiler, debugger, browser, editing, and project management capabilities. ObjectAda version 7.2.1 now fully supports the latest validation suite, ACATS (ACVC) version 2.4. This is the first ObjectAda version to fully conform to this new ACATS version. Along with these technology enhancements, several new task oriented debug capabilities have been added, further enhancing the already powerful debugging capabilities of ObjectAda.

ObjectAda 7.2.1 now supports specific compile time options to optimize code generation for the three supported operating systems. Thus users can further optimize their applications based on the operating system they are targeting. An alternate implementation of the package Ada.Numerics.Generic_Elementary_Functions (RM A.5.1) is provided with ObjectAda 7.2.1. For most applications, the new alternate implementation provides improved execution time performance.

All these capabilities are designed to increase the performance of programs generated with ObjectAda, states Greg Gicca, Product Manager for ObjectAda. All these enhancements bring ObjectAda up to date with the latest related technologies to move ObjectAda onward into the future. [...]

About Aonix

Aonix, a Gores Technology Group company, is a leading international software company with customers drawn from the Global 1000. The Critical Development Solutions (CDS) division produces Software through Pictures(r) (StP), Architecture Component Development(r) (ACD), TeleUSE(r), ObjectAda(r), AdaWorld(r) and Raven. CDS products support the highest criticality levels of software design. The company's Select Business Solutions division provides a comprehensive suite of business software development products comprising Select Component Factory, Select Enterprise(r), Reviewer, JSync, VBSync, and CSync. These tools are supported by a full complement of professional services in addition to the development methodology, Select Perspective. The Enterprise Business Intelligence Solutions division provides products such as Nomad(r) and UltraQuest, and services designed to dramatically simplify access and analysis of mainframe data. Headquartered in San Diego, Aonix operates sales offices throughout North America and Europe in addition to a network of international distributors. For more information, visit www.aonix.com.

Press Contacts: Greg Gicca, Director of Product Management, greg.gicca@aonix.com; additional Product Information: (858) 457-2700, info@aonix.com

Green Hills Software - INTEGRITY RTOS and AdaMULTI SDE Selected for F-35 JSF

URL: http://www.ghs.com/news/211031l.html

Green Hills Software's INTEGRITY(TM) Selected As Operating System For F-35 Joint Strike Fighter

Green Hills to Provide Operating System and Development Environment for Largest Military Procurement in History

Santa Barbara, CA, October 31, 2001 -- Green Hills Software, Inc., today announced that Lockheed Martin will be using Green Hills' INTEGRITY(TM) real-time operating system (RTOS) and AdaMULTI(r) 2000 software development tools to develop software for its Joint Strike Fighter (JSF) aircraft. Lockheed Martin's design for the JSF was selected by the Department of Defense (DoD) in a $200 billion award, the largest in US DoD history. Avionics software developed by Lockheed Martin will run on airborne PowerPC processors operating under the INTEGRITY RTOS.

[For the full text, see URL mentioned above. Some extracts: -- dc]

[...] John Carbone, vice president of marketing at Green Hills [...] continues, "The INTEGRITY RTOS, together with Green Hills' AdaMULTI IDE, and Ada 95/C/C++ compilers provides a complete single-vendor RTOS and development solution for developing real-time mission and safety-critical software systems capable of meeting the security and safety standards of ISO/IEC 15408 (Common Criteria) and RTCA DO-178B. [...]

INTEGRITY RTOS is optimized for safety-critical embedded applications that place a premium on maximum reliability, security, and testability. It features advanced memory protection capabilities, an optional ARINC-653 partition scheduler, dynamic download, task- and system-level debug, a configurable real-time EventAnalyzer, POSIX support, and TCP/IP networking. It is also the first memory-protected real-time operating system to be offered on a royalty-free basis.

INTEGRITY is engineered from the ground up to provide security and determinism. At the lowest level, the kernel employs an object-oriented design and access verification to protect against inadvertent and malicious kernel access problems such as invalid kernel addresses and invalid system call parameters. The kernel design also guarantees bounded computation times by eliminating the need for features such as dynamic memory allocation and heuristic scheduling. Underlying hardware mechanisms are used to provide full system memory protection of all components, including user applications, device drivers, and inter-address space communications. Clocks and timers are protected with access permissions and implemented entirely in software.

INTEGRITY is tightly integrated with Green Hills' AdaMULTI(r) IDE. Together with Green Hills' family of optimizing Ada 95, C, and C++ compilers, AdaMULTI automates all aspects of embedded software development, including editing, source-level debugging, program building, run-time error checking, version control, and code/performance optimization. INTEGRITY also features ISIM, an RTOS simulator that enables programmers to develop and test their code on a PC or workstation without the need for target hardware. INTEGRITY also includes the EventAnalyzer(TM), which enables viewing of system and user events in a graphical display. [...]

More On Green Hills Software

Founded in 1982, Green Hills Software, Inc., is the technology leader for real-time operating systems and software development tools for 32- and 64-bit embedded systems. Green Hills offers INTEGRITY and INTEGRITY real-time operating systems and a family of optimizing Ada 95 and C/C++ compilers. The company's unique INTEGRITY RTOS provides guaranteed resource availability in both time and space domains for the highest reliability and security in a commercial RTOS.

Green Hills' tools support all of the major advanced microprocessor families and target environments, including target simulators, ROM monitors, other commercial and home grown real-time operating systems, and in-circuit emulators (ICE).

Green Hills Software is headquartered in Santa Barbara, CA., and has 11 US offices located in California, Colorado, Illinois, Massachusetts, North Carolina, Texas, Washington, and Florida. European headquarters are located in the United Kingdom, with offices in France, Germany, the Netherlands and Sweden.

For sales information on Green Hills' safety-critical products, including INTEGRITY, call 800-789-9695 or email inquiries to ada-sales@ghs.com.

For More Information Contact: Green Hills Software, Lynn J. Robinson, (805) 965-6044, lynnr@ghs.com; Davis-Marrin Communications, Michelle Ragsdale, (858) 573-0736, michelle@davismarrin.com

OAR - RTEMS Operating System

From: Simon Clubley <simon_clubley@excite.com>
Date: Mon, 10 Sep 2001 12:37:09 GMT
Subject: RTEMS and Ada, Was: Re: Ada OS talk (was: Progress on AdaOS)
Newsgroups: comp.lang.ada

> OAR had an Ada version of the RTEMS operating system a while back. I think they have dropped support for it, but I'm sure the source is available from them. This would probably be a good starting point [for an Ada operating system. -- dc]

The situation's a bit more complicated than that. :-)

OARcorp _do_ support Ada as a development environment for RTEMS. However, some of the BSP's (for example, the i386 BSP) have moved from COFF to ELF.

The current GCC used for GNAT, 2.8.x, generates COFF format binaries. It has been commented on several times in the RTEMS mailing list that OARcorp are eagerly awaiting for ACT to deliver a GNAT for GCC 3.x. :-)

PS: I am not associated with OARcorp in any way apart from wanting to use Ada with RTEMS...

From: joel@OARcorp.com (Joel Sherrill)
Date: 10 Sep 2001 13:39:07 -0700
Subject: Re: Ada OS talk (was: Progress on AdaOS)
Newsgroups: comp.lang.ada

> I'm aware of RTEMS - it might make "A Good Start". My recollection of what it was all about was that it might be a bit limited for a full-up OS. More like an RTK and some device drivers to support an Ada implementation on a bare board.

Correct RTEMS is what used to be called an executive or kernel. It is not as limited as you might think though with major features like TCP/IP, filesystem, about 85% of POSIX 1003.1b, uITRON, and a pSOS+-like API ported to about a dozen CPUs.

> There might be licensing issues as well. I believe it is GPL - don't know the specifics - but it might not be A Good Thing to force any additional work to fall under the GPL. (That's an opinion - others may differ on that.)

RTEMS is GPL'ed with the same type of exception as the GNAT run-time [hence permitting its use in proprietary software -- dc].

It has been a long time but it can be dug up. :) When it was new, there was very little feedback to it. The C implementation got a lot more interest and when the C version was used as the run-time for GNAT, it seemed to satisfy the same goal without duplicating effort.

Joel Sherrill, Ph.D., joel@OARcorp.com; Ask me about RTEMS: a free RTOS, Support Available; Director of Research & Development, On-Line Applications Research, Huntsville AL 35805, (256) 722-9985

From: Simon Clubley <simon_clubley@excite.com>
Date: Mon, 10 Sep 2001 20:25:33 GMT
Subject: Re: RTEMS and Ada, Was: Re: Ada OS talk (was: Progress on AdaOS)
Newsgroups: comp.lang.ada

[About "BSP's (...) have moved from COFF to ELF."]

> Hmm. In my experience, a Board Support Package (BSP) is written in C, or Ada, or (most likely) Assembler. _Not_ object code.

In this case, the RTEMS BSP's are written in C.

> So what does it mean that a BSP has "moved from COFF to ELF"? Maybe the BSP defines the loader, and it needs to know the object file format. I guess that would make sense.

Yes, I was not precise enough here. Your description above is closer, although as I have not yet successfully built RTEMS with Ada support and as my background is VMS and not Unix, I am still a little unsure on the fine details. My understanding is that, amongst other things, a specific BSP's build environment is designed differently depending on if COFF or ELF formats are in use.

[About "GCC used for GNAT, 2.8.x, generates COFF format binaries."]

> This is misleading. GCC, the compiler, does not generate binaries; it generates Assembler code. The system assembler and linker generate binaries. If using the Gnu assembler and linker, they come from the binutils package, not the GCC package. So just because GNAT uses GCC 2.8.1, does not mean RTEMS can't use it with binutils 2.11 (latest on Gnu site). I suppose there may be some upward incompatibilities, but I'd be surprised.

I do know that when I attempted to do just this (with or without the OARcorp supplied patches to the GCC/GNAT/binutils sources), in order to get pre-ELF versions of the i386 BSP working with GNAT, then the build failed in various ways.

[About "awaiting for ACT to deliver a GNAT for GCC 3.x."]

> They don't have to use only the binary distributions from ACT! I would consider it part of a 3rd parties job, to repackage things for exactly this kind of reason.

I never implied that they were waiting for a binary distribution. :-) Like the rest of us, I understand they are waiting for ACT to release a source distribution. OARcorp do optionally build some binary environments for easy installation, but generally RTEMS is built from source and a GNAT cross-compiler (or just GCC if you are not interested in using Ada with RTEMS) is also built from source as part of the installation.

PS: Once again, I am not associated with OARcorp in any way apart from an interest in RTEMS.

From: joel@OARcorp.com (Joel Sherrill)
Date: 10 Sep 2001 13:56:18 -0700
Subject: Re: RTEMS and Ada, Was: Re: Ada OS talk (was: Progress on AdaOS)
Newsgroups: comp.lang.ada

[...] GCC 2.8.1 did support ELF for the Sparc and PowerPc so it has not been that difficult to try those out. In fact, I recently tested 3.13p for the PowerPc and posted ACATS results to the RTEMS users list. They were quite good. The Sparc results were also good although the Sparc backend now makes some Solaris dependent calls. :(

We made RPMs available for GNAT/RTEMS 3.13p targetting the PowerPC and they are at

ftp://ftp.oarcorp.com/pub/rtems/snapshots/ada_tools/gnat-3.13p-2 These support the latest development snapshots of RTEMS.

[See also "GNAT 3.13p Linux RPMs for RTEMS" in AUJ 21.4 (January 2001), p.231. -- dc]

> PS: I am not associated with OARcorp in any way apart from wanting to use Ada with RTEMS...

And although I AM associated with OARcorp, personally I want to see Ada and RTEMS work together also. :)

From: "Michael Garrett" <michaelgarrett@csi.com>
Date: Tue, 11 Sep 2001 22:40:18 -0500
Subject: RTEMS Ada Micro Kernel (Was Ada OS Progress)
Newsgroups: comp.lang.ada

I may not understand the whole picture.... (been in managment for a while) but the original Ada version of RTEMS could be put on top of a hardware abstraction layer, with minimal amount of assembly, (it probably was implemented this way) forming a micro kernel, which could be the foundation of the operating system.

With suitable layers on top of this foundation (GNORT compiled, or no runtime), a runtime could be provided that GNAT could call into, providing a full Ada runtime environment on top of an all Ada kernel. (I'm in over my head but you get the picture).

What I think this would provide is deterministic behavior. If an all Ada OS is to be considered, my opinion is that it should be deterministic to a hard realtime level. This is a lofty goal, but starting with a deterministic all Ada micro kernel is a good start.

QNX is built this way, on top of a small scalable micro kernel. If the goal is an all Ada scalable OS this seems to be an ideal way to leverage the work already done by OAR, the all Ada micro kernel.

Michael C. Garrett, Vice President Research and Development, Medical Research Laboratories, www.mrlinc.com, michaelgarrett@csi.com

Praxis Critical Systems - SPARK Toolset 6.0

From: rod@praxis-cs.co.uk (Rod Chapman)
Date: 28 Nov 2001 10:13:43 -0800
Subject: ANN: SPARK Toolset Release 6.0 now available
Newsgroups: comp.lang.ada, comp.software-eng, comp.realtime

Praxis Critical Systems is pleased to announce the immediate availability of release 6.0 of the SPARK language and the SPARK toolset. [...]

A new edition of the "SPARK Book" by John Barnes is also planned.

Release 6.0 adds significant functionality over previous releases, including:

* Support for "external variables" - these allow the automated modelling of volatile input and output "streams" to and from a SPARK program. Our "INFORMED" design approach has been updated to illustrate the use of external variables.
* Modular types with binary modulus are supported in SPARK95 mode. Bit-wise logical operators are permitted for such types.
* A new "derives null from ..." annotation form allows the declaration of procedures which take parameters, but have no observable effect on any state within the SPARK boundary of a program. This is particularly useful in the construction of data-logging packages, testpoints, diagnostic code and so on.
* VC-Generation in much improved. In particular, fewer hypotheses are generated for most VCs, resulting in faster simplification of those VCs [Verification-Conditions -- dc].
* SPADE Simplifier version 2.00 ships on Windows and Solaris platforms with this release. This includes improved proof tactics for modular expressions, bit-wise logical operators, and inequalities involving enumerated types. The new Simplifier behaves identically on Windows and Solaris.
* A new tool called "SPARKSimp" is supplied on Windows and Solaris. This is a "make" style tool for the Simplifier which assists in the proof of large programs.

This release re-inforces SPARK's position as the leading language subset and static analysis technology for the construction of high-integrity software.

Please email us for more information at sparkinfo@praxis-cs.co.uk or see www.sparkada.com

The SPARK Team, Praxis Critical Systems

Note: The SPARK programming language is not sponsored by or affiliated with SPARC International Inc and is not based on SPARC(tm) architecture.

RainCode Corp. - RainCode for Ada

From: Deborah Torrekens <Deborah@raincode.com>
Date: Wed, 05 Dec 2001 18:37:21 +0100
Subject: Ada User Journal
To: Dirk.Craeynest@cs.kuleuven.ac.be

RainCode for Ada - Press Release - November 2001

RainCode for Ada is a quality control technology that operates on large amounts of existing Ada code, both legacy or during development. RainCode detects, counts, and measures non-trivial things in your Ada code, and it can take any corrective or preventive action in it.

RainCode measures compliance to coding standards based on style or on technical matters (portability); or apply potentially large numbers of automated patches. Such patch strategies address issues such as modularisation, comments generation, etc.

How does it work? RainCode actually reads the Ada source code and builds an annotated parse tree after a fully documented object model. A scripting language can then be used to walk through the parse tree, taking full advantage of the features provided by the object model.

RainCode for Ada performs more than just syntactical analysis; it includes type analysis and a complete access to the tagging information for each and every function, procedure, variable and type in the code.

The scripting language is a dynamically typed Pascal-like language, which has a number of very useful features such as:

* Quantifiers
* Modularity
* Set-based operations (membership, union, difference, etc)
* Access to XML trees through the DOM API
* Pattern matching, which allows you to comfortably express what you are looking for in a program

More information available at www.raincode.com

Deborah Torrekens, Raincode Corp., Rue de l'Autonomie, 1, B-1070 Bruxelles, Belgium, Tel 322 + 522.06.63, Fax 322 + 522.09.30

Rational - Apex 4.0.0c, TestMate 4.0.0, Ada Analyzer 4.0.1, and AXI 4.1.9 for IBM AIX

From: "Greg Bek" <gab@Rational.Com>
Date: Tue, 30 Oct 2001 12:36:23 -0800
Subject: Apex 4.0.0c, TestMate 4.0.0, Ada Analyzer 4.0.1, AXI 4.1.9 for IBM AIX are available by FTP
To: "Apex Announcements" <apex-announcements@Rational.Com>

Rational Apex 4.0.0c, Rational TestMate 4.0.0, Ada Analyzer 4.0.1, and AXI 4.1.9 for IBM AIX are available by FTP.

[In all Rational' URLs below, substitute <apex> by <ftp>/apex, <analyzer> by <ftp>/ada_analyzer, <axi> by <ftp>/axi, <testmate> by <ftp>/testmate, <doc> by <documents/unix>, and <ftp> by ftp://ftp.rational.com/public -- dc]

These releases are pending Generally Available (GA) status as they go through the final steps of the manufacturing process. We anticipate that this will be complete within the next 30 days. Once these releases reach GA status, they will be available for shipping. Until then, they are being provided on this FTP server for immediate access. Follow the links for download and installation instructions. There are 3 ways to do the download. You can use the classic method of the UNIX ftp command, use a web browser, or use Rational's rinstall program to do the FTP download in a user-friendly way. <ftp>/.standard.msgs/install_instructions.html

Product: Rational Apex Version: 4.0.0c Platform: IBM AIX URL: <apex>/releases/aix/apex.4.0.0c Release Note: <apex>/<doc>/release_note.4.0.0c.dir/release_noteTOC.html <apex>/<doc>/release_note.4.0.0c.ps.Z Install Guide: <apex>/<doc>/install_guide.4.0.0c.dir/igTOC.html <apex>/<doc>/install_guide.4.0.0c.ps.Z

Product: Rational TestMate Version: 4.0.0 Platform: IBM AIX URL: <testmate>/releases/aix/testmate.4.0.0 Release Note: <testmate>/<doc>/testmate_release_note.4.0.0.dir/testmate_noteTOC.html <testmate>/<doc>/testmate_release_note.4.0.0.ps.Z

Product: Ada Analyzer Version: 4.0.1 Platform: IBM AIX URL: <analyzer>/releases/aix/ada_analyzer.4.0.1 Release Note: <analyzer>/<doc>/release_note.4.0.1.dir/AA_Release_Note.4.0.html <analyzer>/<doc>/release_note.4.0.1.ps.Z

Product: AXI (Ada/X Interface) Version: 4.1.9 Platform: IBM AIX URL: <axi>/releases/aix/axi.4.1.9 Release Note: <axi>/<doc>/axi_relnotes.4.1.9.aix.dir/relnotesTOC.html <axi>/<doc>/axi_relnotes.4.1.9.aix.ps.Z

Rational - Apex 4.0.0b for Windows NT

From: "Greg Bek" <gab@Rational.Com>
Date: Tue, 13 Nov 2001 13:18:21 +1030
Subject: Apex 4.0.0b, Ada Analyzer 4.0.1 for Windows NT are available by FTP
To: "Apex Announcements" <apex-announcements@Rational.Com>

[...]

Product: Rational Apex Version: 4.0.0b Platform: Windows NT (Intel) Directory URL: <apex>/releases/win/apex_nt.4.0.0b-self_extract File URL: <apex>/releases/win/apex_nt.4.0.0b-self_extract/apex_nt.400b.exe

Common Object Request Broker Architecture (CORBA)

CORBA vs. Distributed Systems Annex

From: "Jean-Pierre Rosen" <rosen@adalog.fr>
Date: Mon, 3 Sep 2001 19:45:12 +0200
Organization: Adalog
Subject: Re: CORBA vs. Distributed Systems Annex?
Newsgroups: comp.lang.ada

> Is there a brief description of the differences between CORBA and DSA? I don't know much about either, and I'm looking for a starting point in deciding which to use.

In a nutshell: DSA is for one (Ada) program, whose execution is distributed over several machines (aka partitions). CORBA is for several, possibly heterogenous, programs communicating together.

J-P. Rosen, http://www.adalog.fr

From: Samuel Tardieu <sam@rfc1149.net>
Date: Mon, 3 Sep 2001 19:45:51 +0200
Subject: Re: CORBA vs. Distributed Systems Annex?
Newsgroups: comp.lang.ada

You can get an article called "CORBA & DSA: divorce or marriage?" from Laurent Pautet, Thomas Quinot and myself published in Ada Europe '99 on http://www.rfc1149.net/biblio

From: Thierry Lelegard <thierry.lelegard@canal-plus.fr>
Date: Tue, 04 Sep 2001 15:59:35 +0200
Organization: CANAL+ Technologies
Subject: Re: CORBA vs. Distributed Systems Annex?
Newsgroups: comp.lang.ada

At the SIGAda'99 conference, there were a number of sessions on the subject. See http://www.acm.org/sigada/conf/sigada99/Sessions.html

Thierry Lelegard, CANAL+ Technologies, 34 place Raoul Dautry, F-75906 Paris Cedex 15, France, Tel +33 1 71 71 54 30, Fax +33 1 71 71 52 08

Selecting CORBA Implementations

From: Nielson Mark S Civ OO-ALC/TISEB <Mark.Nielson@hill.af.mil>
Date: Wed, 24 Oct 2001 14:47:27 -0600
Subject: The November Issue of CrossTalk is now available on-line.
To: Dirk@offis.be

The November 2001 issue of CrossTalk, The Journal of Defense Software Engineering is now available on our Web site at: http://www.stsc.hill.af.mil. The theme of this month's issue is "Distributed Software Development." [...]

If you are in the market for distributed software middleware, Dr. Thomas Croak presents Factors to Consider When Selecting CORBA Implementations. This article can help you choose and fine-tune a Common Object Request Broker Architecture (CORBA) Object Request Broker. It looks at 10 software architecture variability dimensions that cause different behaviors in CORBA. [...]

Ada and Linux

New Linux packages on www.gnuada.org

From: Jürgen Pfeifer <juergen.pfeifer@gmx.net>
Date: Thu, 13 Sep 2001 01:42:28 +0200
Subject: News
To: "GNAT Discussion List" <gnatlist@lyris.seas.gwu.edu>

I've updated www.gnuada.org. There are new RPM releases for GNU/Linux of the following packages:

- XML/Ada (Version 0.6, Release 1)
- AWS (Version 1.0, Release 1)
- GNADE (Version 1.1.5, Release 1)

Please note that on the homepage there is now also a link for Ada on OS/2. John Poltorak asked for that link and he took responsibility to provide the content. If you're an OS2er, stay tuned...

Please note that the spelling for ALT has changed a little bit. ALT = Ada for GNU/Linux Team ;-)

From: Jürgen Pfeifer <juergen.pfeifer@gmx.net>
Date: Tue, 20 Nov 2001 01:19:18 +0100
Subject: News
To: "GNAT Discussion List" <gnatlist@lyris.seas.gwu.edu>

I've uploaded new builds of these packages:

- XmlAda (minor release, fixes an omission in previous RPMs)
- Adasl (new version 1.4)
- Adabindx (new version)

Ada RMPs in Next SuSE distribution

From: Jürgen Pfeifer <juergen.pfeifer@gmx.net>
Date: Wed, 3 Oct 2001 11:59:40 +0200
Subject: ALT RPMs in next SuSE distribution.
To: "GNAT Discussion List" <gnatlist@lyris.seas.gwu.edu>

I just want to let you know that the next release of SuSE Linux (7.3) will contain the ALT packages for GNAT.

Ada and Microsoft

GWindows - Ada 95 Win32 RAD Framework

From: "David Botton" <David@Botton.com>
Date: Thu, 6 Sep 2001 23:57:13 -0400
Subject: GWindows progress
Newsgroups: comp.lang.ada

GWindows has been getting some attention these days.....

What is GWindows you ask?

GWindows is a framework for quickly creating Windows applications the Ada_ way, released with the freedoms of the Modified GPL. It is more then just a comprehensive "thick" binding to Win32 implementing additional extensions to ease development on Windows beyond just making the API accessible. It offers features not found in any other framework or binding such as support for __GUI__ ActiveX controls, full UNICODE support (a simple build switch and GWindows is completely UNICODE) for getting the best performance out of WinNT and Win2K, and _both_ handler (access to subprogram) and inheritance based event models.

[See also "GWindows 0.1 - Win32 RAD Development Environment" in AUJ 21.3 (October 2000), p.170. -- dc]

Currently it is in what I call pre-beta (it needs more work on printing support and there are a few additional methods I would like to add to some of the controls) and should be beta with in weeks. It is fully functional and has been used already by a number of people (including myself) for some decent size projects.

Full database integration is already in the works for the 1.1 version to allow for rapid development of database front ends in Ada (the gap between the power of Ada and the ease of VB is shrinking....) Tutorials and documentation have also been started. (Although, the specs are already well documented and the samples should get you off to a good start.)

Between GWindows and GNATCOM you have bindings (or the equivalent there of) to almost every single facet of Windows. You can "Experience" the ease of development with GWindows by visiting and downloading at http://www.adapower.com/gwindows and joining the GNATCOM mailing list where GWindows is discussed.

A quick list of features available now:

Bound Objects: Windows, Buttons, Default Buttons, Cancel Buttons, Radio Buttons, Check Box Buttons, Three State Buttons, Group Boxes, Combo Boxes, Drop Down Combo Boxes, Drop Down List Boxes, List Boxes, Multiple Selection List Boxes, Edit Boxes, Multi Line Edit Boxes, Rich Text Edit Boxes (both single and multi line), Scroll Bar Controls (and Window scroll bars), Labels, Icon controls, Bitmap controls, Menus (Both window and right click styles), Accelerator Tables, Open and Save File Common Dialogs, Color Common Dialog, Cursors, Carets, Dialogs, Image Lists, AVI Animation Control, Status Bars, Date/Time Controls w/ Pop-Up Calendar, IP Address Control, Progress Control, Up Down Control, Trackbar Control, List View Control, Tree View Control, Tab Control, Tool Tips, Owner Drawn Controls

Other Features: COM Access to Rich Edit - Text Object Model (TOM), Scroll Panels, MDI Support, Keyboard support, GDI Drawing (Windows, Bitmaps and the Printer), Garbage collecting on dynamically created GWindows objects, Listener/Handler (access to subprogram) and OO Event Models, Prefab Events, ActiveX controls, Custom controls, Create windows from dialog resources, Use windows as dialogs, Message Boxes / Beeps, File Drag and Drop, Z-Order control, Window Docking, Examples (Web Browser, PDF Viewer, Embedded Tcl/TK widgets...), Registry access, Tab controls with child window support, ANSI and UNICODE support, more...

From: "David Botton" <David@Botton.com>
Date: Sun, 9 Sep 2001 23:58:03 -0400
Subject: GWindows Tutorials
Newsgroups: comp.lang.ada

Ever wish you could be programming Windows applications with Ada? Maybe you thought it was too difficult or the learning curve to high...

GWindows makes it easy and learning to program GWindows just got easier!

So far, 11 short tutorials and more on the way. Each introducing another simple facet of the power of the GWindows frame work.

Go ahead take a look: http://www.adapower.com/gwindows/user_guide.html then take a look at the many sample programs included in the distribution to get a little more depth.

GWindows should be going Beta this week after I complete my current review of the code.

What is GWindows? Take a look at http://www.adapower.com/gwindows to get an idea.

From: "David Botton" <David@Botton.com>
Date: Tue, 11 Sep 2001 00:57:52 -0400
Subject: Easy databases with Ada
Newsgroups: comp.lang.ada

I've added a new package gwindows-databases that allows easy access to OLEDB and ODBC databaes via ADO.

I hope to add very soon events for changes to fields, record movement, etc. and databound controls. Yup, you will be able to link text boxes, etc. to fields in the database and automaticly have them update when you move around the queries/tables, edit the field values, etc. The power of Ada with the ease of VB (or Delphi ;-)

Here is a simple non-GUI example of using the databases (from Tutorial 18): [30 lines deleted, see online tutorial. -- dc]

Couldn't get any easier! All this and more in the current release on the GWindows page, http://www.adapower.com/gwindows [...]

From: David Botton <David@botton.com>
Date: Thu, 13 Sep 2001 14:50:46 -0400
Subject: Re: printing
To: team-ada@acm.org

[In response to a question about printing from an Ada 95 program on Windows. -- dc]

> If you want to draw lines and use fonts, you have to access the Win32 GUI functions, via Windex or Claw or GtkAda, and write to a printer device instead of a screen device.

You can also get the Beta of GWindows that includes printing support and more at http://www.adapower.com/gwindows. Tutorial 10 covers printing using GWindows and can be seen on-line at http://www.adapower.com/gwindows/user_guide.html.

From: "David Botton" <David@Botton.com>
Date: Thu, 13 Sep 2001 15:34:51 -0400
Subject: GWindows Announcement
Newsgroups: comp.lang.ada

Announcing the first beta release of GWindows

The Ada 95 Win32 RAD Framework

September 13, 2001

GWindows, the Professional Open Source Ada 95 Win32 RAD Framework, introduces for the first time to Ada programming a comprehensive rapid application development framework spanning GUI, Database and Active X integration. It brings AdaPower ;-) to programming domains that up until now are dominated by VB and Delphi.

GWindows includes extensive bindings to the Windows GUI including support for common controls and dialogs, printing, and owner drawn extensions to controls. In addition, GWindows adds a number of new controls, keyboard support, multiple models of event handling, Active X controls, support for

creating dialogs and windows from resource files, dynamic garbage collected windows, Window docking, non-GUI bindings, database support, _database bound controls_, and much more!

GWindows builds as either ANSI or UNICODE (a first for Ada!) for internationalization and performance boosts on Windows NT, 2000, and XP. GWindows is tightly integrated with GNATCOM, the Ada 95 COM/DCOM/COM+ Development Framework and Tools opening every facet of the Windows platforms to Ada 95 development. Never again will the cries be heard, "but there are no bindings" on the Windows platform!

GWindows is designed to take advantage of Ada's unique combination of features rich typing mechanisms. It is not a thick binding to an underlying C interface, but a complete framework that takes advantage of Ada at every level.

GWindows is already being used for production products by large companies to small personal projects. The final release of GWindows will follow the beta period. During the beta period bug fixes, if any will be made available and reported on the mailing list.

GWindows is being made available under the GNAT modified GNU GPL used by GNAT's runtime library making it available for use in both GPL and proprietary applications.

For more information on GWindows, to view the on-line documentation, and to download the product, please visit http://www.adapower.com/gwindows.

Tutorials for GWindows are available at: http://www.adapower.com/gwindows/user_guide.html

Information on the public version of GNATCOM can be found at http://www.adapower.com/gnatcom. Professional support for GNATCOM is available from Ada Core Technologies, Inc. Please contact report@gnat.com for a no-cost GNATCOM evaluation package.

From: "David Botton" <David@Botton.com>
Date: Sun, 7 Oct 2001 12:01:18 -0400
Subject: GWindows Precompiled
Newsgroups: comp.lang.ada

GWindows the Open Source Win32 RAD Ada 95 GUI Development Framework now has a precompiled version available to ease installation on Windows 9X boxes. Information is available at http://www.adapower.com/gwindows

Be sure to check out the GWindows tutorials to Jump Start you in to GUI and Database front end programming on Windows with Ada 95.

GWindows - The Power of Ada 95 with the Ease of VB and Delphi

Adobe's SVG Engine and Ada 95

From: "David Botton" <David@Botton.com>
Date: Sat, 8 Sep 2001 23:41:58 -0400
Subject: SVG and Ada 95 - Re: [ANNOUNCE] XML/Ada 0.6 released
Newsgroups: comp.lang.ada

> I can of course take advantage with Ada already on Win32 of Adobe's SVG engine using GNATCOM, but it would be very nice to have a cross platform version.

I have created a stand alone SVG viewer application in the latest release of GWindows (gwindows\samples\svg) using GWindow's ActiveX support.

After installing the latest gwindows (don't forget GNATCOM must be installed first) and after downloading and installing the Adobe SVG component (http://www.adobe.com/svg/viewer/install/main.html) you can go to the directory gwindows\samples\svg and do a make. Once you open svg_demo you can open the file GWindows.svg

You can take a look at the bindings generated by GNATCOM to see that the SVG control gives you increadible power over each element in the SVG file. If I find the time, perhaps I'll see if I can come up with a demo of doing some interesting interactions between Ada code and SVG elements.

Returning from the Dark Side to Ada

From: David Botton <David@botton.com>
Date: Sun, 7 Oct 2001 12:10:26 -0400
Subject: Returning from the Dark Side to Ada
To: team-ada@acm.org

The question is, now that it is possible to create powerful Win32 GUI applications and Database Front Ends in Ada 95 using GWindows will those Ada 95 people who defected to the Dark Side for the empty promises of VB return to see the light.....

http://www.adapower.com/gwindows

GWindows the Open Source Win32 RAD Ada 95 GUI Development Framework now has a precompiled version available to ease installation. Information is available at http://www.adapower.com/gwindows

Be sure to check out the GWindows tutorials to Jump Start you in to GUI and Database front end programming on Windows with Ada 95.

GWindows - The Power of Ada 95, VB and Delphi combined.

At least one GUI builder is already in development and others are being discussed. It is my intention to pound on VB and Delphi lists once at least one of them is available as the Open Source Alternative to VB and Delphi... I hope everyone is ready ;-)

From: Frank Manning <frankmanning@earthlink.net>
Date: Mon, 8 Oct 2001 21:36:48 -0700
Subject: Re: Returning from the Dark Side to Ada
To: team-ada@acm.org

> It is my intention to pound on VB and Delphi lists once at least one of them is available as the Open Source Alternative to VB and Delphi... I hope everyone is ready ;-)

In the case of VB, you probably couldn't pick better timing, with VB.NET coming up, IMHO. Where I work, we do a lot of VB development, and we're facing the rather daunting prospect of migrating to VB.NET, which has significant incompatibilities relative to VB6.

Although there are translation tools available, even Microsoft apparently takes the position that migration will involve more rewrite than porting. Plus there's all the new stuff we'd have to learn -- managed/unmanaged code, assemblies, delegation, MSIL, CLR, etc. Can't say I'm overjoyed at the prospect of transitioning from one proprietary language to yet another proprietary language.

If migration is going to be such a hassle, why not migrate to an ISO language? Like, well, you-know-what?

While we're on the subject, is there an Ada guide for VB programmers that is similar to Simon Johnston's excellent guide for C/C++ programmers that's posted at adahome.com?

Frank Manning, Tucson, AZ

From: "S. Ron Oliver" <sroliver@csc.calpoly.edu>
Date: Tue, 9 Oct 2001 07:34:03 -0600
Subject: Re: Returning from the Dark Side to Ada
To: team-ada@acm.org

I have been following this thread with interest.

I believe David's original point that many of the "arguments" for using VB, Delphi, etc., and thus for NOT using Ada, are completely invalidated. (Actually, I think they have been for a long time, but the work David and others have done recently makes this so clear even a C programmer will probably understand. :)

There is one key ingredient remaining, however, to get people to use these wonderful tools (and Ada) - getting the word out. Remember, best kept secrets...

More importantly, the BEST way to "get the word out" is to put together good training courses and get them presented in a forum where practicing and would be "programmers" can readily take advantage of them. I don't have any

figures, but I suspect that a fairly high percentage of these practicing "programmers" would never have "cut a line of code", if they hadn't been able to attend a training session that showed them how easy it is to do so much by imitating some good examples.

S. Ron Oliver, semi-retired professor of Computer Science and Computer Engineering, www.csc.calpoly.edu/~sroliver

NT_Console Package

From: Jeffrey Carter <jeffrey.carter@boeing.com>
Date: Wed, 7 Nov 2001 16:37:52 GMT
Organization: The Boeing Company
Subject: Re: Clear screen
Newsgroups: comp.lang.ada

> Do you mean to clear a console window? If so I would guess it is one of the ANSI Escape Codes. Have a look here...

Except that those don't work on WinNT. For that you need Jerry Van Dijk's NT_Console package available from http://home.trouwweb.nl/Jerry/packages.html#CONSOLE

References to Publications

George Romanski - "The Challenges of Software Certification"

From: Nielson Mark S Civ OO-ALC/TISEB <Mark.Nielson@hill.af.mil>
Date: Wed, 29 Aug 2001 09:39:32 -0600
Subject: The September 2001 Issue of CrossTalk is now available on-line.
To: Dirk@offis.be

The September 2001 issue of CrossTalk, The Journal of Defense Software Engineering is now available on our Web site at: http://www.stsc.hill.af.mil. [...] The theme of this month's issue is "Avionics Modernization."

[...] stay informed on "The Challenges of Software Certification" in George Romanski's article. Lastly, Lockheed Martin shares its lessons learned in modernization of the C-130 in "Avionics Modernization and the C-130J Software Factory" by Richard Conn, Stephen Traub, and Steven Chung. [...]

[Direct URL to the former: http://www.stsc.hill.af.mil/crosstalk/2001/sep/romanski.pdf -- dc]

DDC-I Online News

From: JC <jcdk@ddci.com>
Date: Fri, 31 Aug 2001 10:26:48 -0700 (MST)
Subject: Real-Time Industry Updates - News from DDC-I
To: T8DK Online News <jcdk@ddci.com>

DDC-I Online News August 2001, Vol. 2, Num. 6 A monthly news update dedicated to DDC-I customers & registered subscribers.

This Month:

* Ada Stands Ready for the 21st Century. Invented for, but no longer required by, the military, Ada is a clear winner in the era of COTS because of its inherent safety-critical reliability, modularity and excellent lifecycle underpinnings. This article written by Joyce Tokar, V.P. of Technology for DDC-I, was published in the May 2001 issue of the COTS Journal. Read how Ada is finding its way into numerous nonmilitary applications.

[...] For the complete newsletter, go to http://www.ddci.com/news_vol2num6.shtml [...]

From: JC <jcdk@ddci.com>
Date: Thu, 27 Sep 2001 14:48:53 -0700 (MST)
Subject: Real-Time Industry Updates - News from DDC-I
To: V8 Sept 2001 Online News - DK <jcdk@ddci.com>

DDC-I Online News September 2001, Vol. 2, Num. 7 A monthly news update dedicated to DDC-I customers & registered subscribers.

This Month:

* Patterns-One Way to Solve the Reuse Problem An introduction to patterns and why they are important for software development.
* Support for IDE Hard-disks from the DACS-80x86 Cross Compiler System. DACS-80x86 now supports IDE Hard-disks. Read more about this hardware independent interface system which provides basic functionality for file manipulation such as open, close, read and write.

[...] For the complete newsletter, go to http://www.ddci.com/news_vol2num7.shtml [...]

From: JC <jcdk@ddci.com>
Date: Tue, 23 Oct 2001 14:46:33 -0700
Subject: Real-Time Industry Updates - News from DDC-I
To: W8 - DK Oct 2001 Online News <jcdk@ddci.com>

DDC-I Online News October 2001, Vol. 2, Num. 8 A monthly news update dedicated to DDC-I customers & registered subscribers.

This Month:

* Where in the World is SCORE Rabbit? Enter to win a $50.00 gift certificate to Amazon.com. Each month a new contest begins. Check it out and win money!!!
* The Vasa: A Disaster Story with Software Analogy The story of an engineering tragedy -- the sinking of the Swedish warship, Vasa, on her maiden voyage on August 10, 1628. The Vasa disaster was an obvious failure but no one did anything wrong! Read this fascinating story and see the software parallels.

[...] For the complete newsletter, go to http://www.ddci.com/news_vol2num8.shtml [...]

Marc A. Criley - "A Socket-Based Manifestation of Streams"

From: "Marc A. Criley" <mcqada@earthlink.net>
Date: Tue, 04 Sep 2001 12:02:27 GMT
Organization: Quadrus Corporation
Subject: Re: adasockets and adatypes
Newsgroups: comp.lang.ada

> Does anyone know a way of sending an Ada type (especially a record of string and enumerated types) down a socket so it can be received by another ada program which recognises the type....

Certainly! See my article in Ada Letters: http://www.acm.org/sigs/sigada/ada_letters/issues/june2001/socket_streams.pdf

Ada Book Recommendations

From: Thomas Smets <tsmets@altern.org>
Date: Mon, 17 Sep 2001 16:34:27 +0200
Organization: Brutele sc, 29, rue de Naples, B1050 Bruxelles, Belgium
Subject: book
Newsgroups: comp.lang.ada

I've been on www.bn.com (Barnes & Nobles) to look for a good Ada book. They all seem decent but I've no idea which I should choose from the list http://shop.barnesandnoble.com/booksearch/results.asp?WRD=ada+programming Would someone have a tip for me? I'm working most of my time on the following OSes: Linux (MDK or Slack), WinNT.

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Mon, 17 Sep 2001 11:53:36 -0400
Subject: Re: book
Newsgroups: comp.lang.ada

You will find lots of book resources at: http://www.adapower.com/. There are descriptions and links there for a variety of books. Check it out.

A favorite of mine for a "get started in Ada quickly" book is "Ada Essential: Overview, Examples and Glossary" which is bookmarked at Adapower. (For your convenience: http://www.learnada.com/)

For a more in-depth view, I rather like: "Programming in Ada 95" by John Barnes, but many others have recommended "Ada as a Second Language" by Norm Cohen. (Sorry. Never read that one.) Both have good reputations as good, general-purpose texts.

Other books you can investigate at AdaPower may deal with Ada from the perspective of a special interest (OOP, Realtime, etc.). Look over the bibliography to see what might best suit your needs.

From: "David Botton" <David@Botton.com>
Date: Mon, 17 Sep 2001 12:01:16 -0400
Subject: Re: book
Newsgroups: comp.lang.ada

It is more about your style and level of learning. Your platform doesn't matter much, Ada is not compiler/platform dependant as all implementations tend to provide the standard as a minimum (that should be refreshing if you have C++ experience :-)

You can find information about books at: http://www.adapower.com/books. Reviews at: http://www.seas.gwu.edu/~mfeldman/ada95books.html. On line resources in general at: http://www.adapower.com. Reference material at: http://www.adapower.com/ref.

As for platform specific information, I suggest starting with: http://www.gnuada.org for Linux compiler and links to resources for Linux, http://www.adapower.com/windows for links to various Windows bindings.

From: Darren New <dnew@san.rr.com>
Date: Mon, 17 Sep 2001 16:07:59 GMT
Subject: Re: book
Newsgroups: comp.lang.ada

I really liked "Ada as a Second Language." Very well organized. Very useful if you already know C++, COBOL, and/or Fortran 9x. Well-indexed as well. Extremely dense, in the sense that as you get farther into it, it may take several long seconds to figure out what a single sentence means, since if you haven't internalized all the Ada terminology, you have to translate from "classwide types with derived types containing unconstrained arrays with limited components ..." to what that actually means in English. :-)

It is not specific to any particular OS or compiler, however. (Which may be good or bad, depending on your needs.)

Darren New, San Diego, CA, USA (PST).

Roderick Chapman - "SPARK and Abstract Interpretation"

From: rod@praxis-cs.co.uk (Rod Chapman)
Date: 21 Sep 2001 06:56:14 -0700
Subject: ANNOUNCE: New white paper available on www.sparkada.com
Newsgroups: comp.lang.ada

A new white paper entitled "SPARK and Abstract Interpretation" is now available for download on www.sparkada.com. The introduction reads:

"Recently, there has been significant interest in the use of Abstract Interpretation (AI) technology in the static analysis of critical software. A number of AI-based tools exist, but some of their marketing suffers from a level of hyperbole that is at best optimistic, and at worst somewhat irresponsible.

There have also been some attempts to compare AI-based static analysis tools with the analysis implemented by the SPARK language and the SPARK Examiner toolset. The aim of this white paper is to dispel some of the common myths and to avoid potential confusion with customers."

I'm sure many readers of c.l.a might be interested in this.

Rod Chapman, SPARK Team, Praxis Critical Systems, sparkinfo@praxis-cs.co.uk

From: Manuel Carro <boris@lml.ls.fi.upm.es>
Date: 24 Sep 2001 15:24:54 +0200
Organization: Computer Science Department, Technical U. of Madrid, Spain
Subject: Re: ANNOUNCE: New white paper available on www.sparkada.com
Newsgroups: comp.lang.ada

> Do you have any links to description of what Abstract Interpretation is - i.e. what is it based on?

Basically it is a general means to analyze programs by mapping them into an abstract value space. As a simple example, all numbers can be mapped to either 0, positive, or negative, and the builtin operations be redefined accordingly, i.e.,

x * y = y * x; (0) * _ = (0); (+) * (+) = (+); (+) * (-) = (-); (-) * (-) = (-); ....

The program is then run in the abstract domain, possibly several times, until the information concerning the program does not change. Then one might be able to infer that a variable is, e.g., always positive at some point. The good point is that having a finite abtract domain (with some mathematical properties) ensures termination of the analysis. The bad point is that, of course, information is lost both with respect to the actual program (I know that something is positive, but that does not help me to get rid of a "X > 3" test), and with respect to the abstract domain itself (i.e., I might end up with a variable which has "any value").

The abstract domain should be carefully chosen to reflect the properties one wants to study. The nice thing is that the analysis algorithm can be made (in principle) generic and be used with any abstract domain.

There has been a lot of work in abstract interpretation in logic and declarative languages.

Manuel Carro, DLSIIS, e-mail: mcarro@fi.upm.es, http://lml.ls.fi.upm.es/~boris, Phone +34 91 336-7455, Fax +34 91 336-7412

From: "Ken Garlington" <Ken.Garlington@computer.org>
Date: Tue, 25 Sep 2001 17:42:40 GMT
Subject: Re: ANNOUNCE: New white paper available on www.sparkada.com
Newsgroups: comp.lang.ada

There is also a brief description at http://www.polyspace.com/abstract.htm

Availability of "High Integrity Ada: The SPARK Approach"

From: rod@praxis-cs.co.uk (Rod Chapman)
Date: 19 Oct 2001 05:52:40 -0700
Subject: ANNOUNCE: High Integrity Ada: The SPARK Approach availability
Newsgroups: comp.lang.ada

It has come to our attention that many people are experiencing significant difficulty, especially in the USA, in obtaining the "SPARK Book" by John Barnes.

The book is definitely in print, and is available (we have about 40 copies here in the office for courses...), despite what is said by the various on-line booksellers and their databases.

If you're having trouble obtaining a copy, please contact us directly at sparkinfo@praxis-cs.co.uk and we will seek to rectify the situation. Knowing how many people are in this situation will be useful for us to pass on to our publisher.

John English - "Ada 95: The Craft of Object-Oriented Programming"

From: John English <je@brighton.ac.uk>
Date: Sun, 23 Sep 2001 00:00:21 +0100
Organization: University of Brighton
Subject: ANNOUNCE: Online Ada textbook
Newsgroups: comp.lang.ada

Since my book "Ada 95: The Craft of Object-Oriented Programming" is now out of print following Prentice Hall being

swallowed up by the all-enveloping Pearson group, I have reacquired the copyright and released it online at http://www.it.bton.ac.uk/staff/je/adacraft/ in HTML format. It can also be downloaded for offline use.

I've fixed the errata from the print edition (I think) but there will no doubt be others that I've missed (or added). Please let me know if you spot anything that needs fixing... Enjoy!

John English, Senior Lecturer, Dept. of Computing, University of Brighton, je@brighton.ac.uk, http://www.it.bton.ac.uk/staff/je, non-profit CD for CS students: see http://burks.bton.ac.uk

From: John English <je@brighton.ac.uk>
Date: Fri, 28 Sep 2001 17:53:48 +0100
Organization: University of Brighton
Subject: Re: ANNOUNCE: Online Ada textbook
Newsgroups: comp.lang.ada

> It would make a very nice part of a student/beginner/hobbyist "kit" - having the HTML & possibly PDF available on a CD with a compiler, etc. Getting the pieces assembled in one place and making it easy for the beginner to get started with Ada would help a lot. This on-line book is an excelent addition to all the pieces and one that has been, to some extent, missing.

See my .sig for a link to the BURKS set of CDs. This set is priced at £7.50 for 4 CDs (a DVD edition will also be available shortly), and it includes my book and Mike Smith's book, the RM and Rationale, the old FAQs and Lovelace, as well as GNAT, GNATCOM, my GnatIDE, AdaGIDE, GVD, JEWL, GTk, TASH, ...

The whole lot is also online -- the Ada page is here: http://burks.bton.ac.uk/burks/language/ada/

It also includes tutorials, reference material and compilers for about 20 other languages, a dictionary of computing, a complete set of RFCs, many of the W3C specifications, lots more tutorial and reference material, and a copy of Mandrake Linux. Would this do, do you think? :-)

From: John English <je@brighton.ac.uk>
Date: Tue, 13 Nov 2001 15:54:29 +0000
Organization: University of Brighton
Subject: ANNOUNCE: Online Ada textbook -- update
Newsgroups: comp.lang.ada

A new version of my online textbook, "Ada 95: The Craft of Object Oriented Programming", has now been uploaded to the website at http://www.it.brighton.ac.uk/staff/je/adacraft/. This version corrects a number of glaring errors, both in the text and in the formatting.

Many thanks to those who sent me corrections, and in particular to Tad Ashlock and Jeffrey Cherry who each found an astonishing number of bloopers (with very few in common!)...

Information about HRT-HOOD

From: "Jean-Pierre Rosen" <rosen@adalog.fr>
Date: Fri, 9 Nov 2001 20:16:23 +0100
Organization: Adalog
Subject: Re: Methodology: HRT HOOD
Newsgroups: comp.lang.ada

[In response to a request for information about HRT-HOOD, an old URL of the "HOOD Method Home Page" was given. -- dc]

Thanks for the plug :-), but please use http://www.adalog.fr/huf. The other address is valid, but may change any time.

From: Stephen Leake <stephen.a.leake.1@gsfc.nasa.gov>
Date: 09 Nov 2001 12:13:09 -0500
Organization: NASA Goddard Space Flight Center
Subject: Re: Methodology: HRT HOOD
Newsgroups: comp.lang.ada

Well, this is not about HOOD directly, but try: http://www.tni.fr/tni/offre/stood/index.eng.html. It's a tool for modelling HOOD designs. They also have information about HOOD in general.

From: Michal Nowak <vinnie@inetia.pl>
Date: Fri, 09 Nov 2001 19:05:50 +0100
Subject: Re: Methodology: HRT HOOD
Newsgroups: comp.lang.ada

> Does anyone know of any good websites that have information about HRT-HOOD?

Following a link from [http://www.adalog.fr/huf] I came to ESA home page at http://www.estec.esa.nl/wmwww/WME/oot/. From there it is possible to download very useful documents about HOOD method. That was just in the case if you miss this link, so it may not provide any help if you were there already.

Mikem, Mike Nowak, vinnie@inetia.pl, http://www.geocities.com/vinnie14pl

German Ada Programming Books

From: Georg Bauhaus <sb463ba@l1-hrz.uni-duisburg.de>
Date: Sun, 11 Nov 2001 23:41:56 +0000 (UTC)
Subject: Re: [Newbie] Searching for good german programming guide in Ada95
Newsgroups: comp.lang.ada

> I'm searching for a good programming guide in the internet. All I found was the Lovelace Tutorial... But no german one to see...

[As the replies are obviously mainly of interest for our German-speaking readers, I've kept the responses in German. -- dc]

Manfred Nagl: Softwaretechnik mit Ada 95 (auf 95 achten, es gibt auch eine alte Auflage)

Diana Schmidt: ... Ada, ... objektorientierten Standards (kenne ich aber nicht)

Es gibt eine Reihe aeltere Ada 83 Buecher, die vielleicht hilfreich sind, in vielen Buechereien.

Und keinesfalls sich die Gelegenheit entgehen lassen, fortgesetzt Englisch zu lernen :-) (which means: by all means keep on learning English (I am talking to myself))

From: Alfred Hilscher <Alfred.Hilscher@icn.siemens.de>
Date: Mon, 12 Nov 2001 11:50:00 +0100
Organization: Siemens AG
Subject: Re: [Newbie] Searching for good german programming guide in Ada95
Newsgroups: comp.lang.ada

Erfolgreich programmieren mit Ada. Unter Berücksichtigung des objektorientierten Standards. von D. Schmidt

I have it, and I like it.

And: Ada, eine Einführung von K.P. Kratzer

Online Ada Books

From: Preben Randhol <randhol+abuse@pvv.org>
Date: Mon, 12 Nov 2001 19:02:23 +0000 (UTC)
Organization: Norwegian university of science and technology
Subject: Re: function to generic
Newsgroups: comp.lang.ada

> [...] how it works exactly? I try to find out from ref guides, but I don't find it. :((

Don't read ref. books, read a text on Ada book. Here are some online.

http://www.it.bton.ac.uk/staff/je/adacraft/
http://burks.bton.ac.uk/burks/language/ada/ada95.pdf
http://www.adapower.com/learn/adadistilled.html

From: "David Botton" <David@Botton.com>
Date: Thu, 15 Nov 2001 09:00:09 -0500
Subject: Re: pointers in Ada
Newsgroups: comp.lang.ada

I know of at least seven books on line for Ada.

http://www.adapower.com/learn

* Ada Distilled
* Ada 95: The Craft of Object-Oriented Programming
* Object Oriented Programming in Ada 95

* The Big Online Book of Linux Ada Programming
* Ada in Action
* Introducing Ada 95
* LAW - Lear Ada on the Web = Ada: A Developmental Approach

The top 3 are new, but the other four have been with us for some time. BTW, if you count the RM, Rationale and Style Guide that makes _10_ books on-line. Plus we have Quick Reference Cards and more :-) (http://www.adapower.com/ref)

Java

Generating Ada Package Specifications for Java Class Files

From: David Emery <demery@cox.rr.com>
Date: Sat, 06 Oct 2001 00:55:03 GMT
Subject: Java to Ada...
Newsgroups: comp.lang.ada

> ... Last I heard, the author was working on an option to produce Ada source files from the class files.

Karl Nyberg and I did a paper on this for Ada-Europe a couple of years ago. There's one really significant problem with generating Ada package specs from class files, and that is that the parameter names are missing. Otherwise, most of what you'd want to know about a class can be figured out from the class file, which is kind of neat.

By the way, the absence of named parameter notation is #2 on my list of big mistakes in the Java language. #1 is the lack of separation of spec and body.

[See "Automating the Ada Binding Process for Java - How Far Can We Go?", by David E. Emery, Robert F. Mathis, and Karl A. Nyberg, in L. Asplund (Ed.): "Reliable Software Technologies - Ada-Europe'98", 1998 Ada-Europe International Conference on Reliable Software Technologies, Uppsala, Sweden, June 1998. Proceedings, Springer LNCS 1411. Abstract at http://link.springer.de/link/service/series/0558/tocs/t1411.htm. -- dc]

Ada Inside

Joint Strike Fighter

From: Richard Riehle <richard@adaworks.com>
Date: Sun, 28 Oct 2001 20:55:53 -0800
Organization: AdaWorks Software Engineering
Subject: Joint Strike Fighter
Newsgroups: comp.lang.ada

Now that LMCO has been awarded the contract for JSF, does anyone know if the original plan to do the software in Ada remains unchanged?

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Mon, 29 Oct 2001 10:02:24 -0500
Subject: Re: Joint Strike Fighter
Newsgroups: comp.lang.ada

I used to work on the JSF engine control and that was done in Ada and I can't think of a good reason Pratt & Whitney would want to switch and lose all that verification work. I'd have to verify it with one of my former associates [...] but I would doubt that after years of development and testing anyone would want to throw out all the code and start over and have to once again flight certify the control.

I wouldn't know about all of the other avionics in the JSF. That would be a *lot* of systems. Are there any in particular you had in mind?

From: Paul A Storm <paul.a.storm@lmco.com>
Date: Mon, 29 Oct 2001 09:55:13 -0800
Subject: Re: Joint Strike Fighter
Newsgroups: comp.lang.ada

Ada use is alive and well here at LMCO. ;-)

From: dirk@cs.kuleuven.ac.be (Dirk Craeynest)
Date: 2 Nov 2001 21:03:50 +0100
Organization: Ada-Belgium, c/o Dept. of Computer Science, K.U.Leuven
Subject: Re: Joint Strike Fighter
Newsgroups: comp.lang.ada
Summary: Green Hills' AdaMULTI selected

Check out the recent news at http://www.ghs.com/news/211031l.html. [...]

[See "Green Hills Software - INTEGRITY RTOS and AdaMULTI SDE Selected for F-35 JSF" in this AUJ issue. -- dc]

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Sun, 11 Nov 2001 10:24:28 -0500
Subject: Re: JSF Avionics Software
Newsgroups: comp.lang.ada

In my most recent conversations with colleagues still at Pratt, the word has been that Pratt is still using Ada for the engine control software and has no intention of changing that any time soon. [...]

VSTOL Predecessor of Joint Strike Fighter

From: ian0kerr@my-deja.com (Ian)
Date: 1 Nov 2001 03:38:12 -0800
Subject: Re: Joint Strike Fighter
Newsgroups: comp.lang.ada

> To be generous, there may be issues surrounding the availability of tools and utilities. There is a lot of stuff available for C++ that you might have a hard time duplicating in Ada.

Working currently on the VSTOL predecessor of JSF I don't find anything that we need from Ada and the available tools that C++ would provide.

[Clarified in a subsequent message as:] Ada provides us with everything we need. C++ does not have an advantage for us.

Our process, (on a different project) is described in FULL in:

"GENESYS: An Application of OO technology to Aircraft Display Systems", Neil Davidson, BAE Systems Avionics Ltd, presented in "Symposium on Reliable Object-Oriented Programming", at Institution of Electrical Engineers, Savoy Place, London, 24th October 2001.

Except we use a substitute for Labview. We don't have a problem recruiting Ada experienced software engineers and then practically everyone who wants it gets training. I have done courses on statecharts, UML, UML to Ada95 code generation, Test instrumentation tools for Ada and Ada95 updates.

I don't think it is that difficult to do avionics in Ada95 if you already know C++ properly (assuming some knowledge of SW Engineering principles). On a large project there is a lot of opportunities for mentoring from other more experienced team members.

Indirect Information on Ada Usage

[Extracts from job-ads and other postings illustrating Ada usage around the world. -- dc]

From: Lionel Draghi <Lionel.Draghi@free.fr>
Date: Wed, 29 Aug 2001 20:55:23 GMT
Subject: [Emploi] SchlumbergerSema
Newsgroups: fr.comp.lang.ada

[Extracts translated from French: -- dc]

SchlumbergerSema [...] for our defense department we are looking for junior and senior Ada development engineers. You will be responsible for the implementation of models (coding, unit tests) for a war-game simulation in an object-oriented architecture. [...] Significant Ada 95 experience is indispensable, with a good knowledge of the object-oriented mechanisms of the language. [...]

From: Patrice Serange <pserange@alten.com>
Date: Fri, 31 Aug 2001 09:50:41 +0200
Subject: Re: Ada-related job announcements (was: Fwd: New job opening)
To: Dirk.Craeynest@cs.kuleuven.ac.be

[At URL http://www.cs.kuleuven.ac.be/~dirk/ada-belgium/jobs/ Ada-Belgium maintains web-pages with Ada-related job offers for Belgium. See also "Ada Jobs Web Pages" in AUJ 21.2 (July 2000), pp.97-98. -- dc]

Ada Software Engineers & Project Managers (Belgium)

Within our embedded competence center, we are looking for engineers and project managers specialized in Ada design and development. Projects are varied and could reach managerial responsabilities depending on your experience. The sectors involved are: aeronautic (design of avionic equipment) or railway. The evolution of this role can be international depending on your motivation. The knowledge of real-time constrained methodologies of development are an advantage, as well as experience in aeronautic or military standards. [...]

From: Christophe Le Bris
Date: Fri, 7 Sep 2001 19:19:32 +0200
URL: <ada-france>/2001-September/000365.html

[Ada-France posts Ada-related job announcements to the fr.comp.lang.ada newsgroup and to Ada-France's mailing list. The latter are available at http://www.ada-france.org/pipermail/ada-france (abbreviated in URLs here as <ada-france>); extracts from a discussion about projects where Ada is used, are included below, translated from French: -- dc]

- Ariane 5 (both on-board and ground) - earth observing satelites (SPOT 5, Helios 2)
- all flight software for geostationary satelites based on the SpaceBus platform (including the new generation now in development)
- all flight software for the small satelites in low orbit based on the Proteus platform.

From: Xavier Gandibleux
Date: Sat, 08 Sep 2001 12:39:09 +0200
URL: <ada-france>/2001-September/000382.html

Add transport-related activies as well (railway and aviation). [...]

From: Culos Alain
Date: Mon, 10 Sep 2001 09:46:34 +0100
URL: <ada-france>/2001-September/000368.html

[...] As well as in all air traffic control systems delivered by Thomson (now Thales). Thales is one of the world leaders in this domain.

Add also the supervision of the Syracuse II system (French inter-army communications). [...]

From: Francois Gody <francois.gody@amaltheus.com>
Date: Mon, 10 Sep 2001 15:00:58 +0200
URL: <ada-france>/2001-September/000370.html

- EADS (Aerospatiale, Airbus, ...)
- Thales (several projects both military as well as civil)
- CNES (several satelites and scientific missions)
- the space station (ISS) has most software for the European Columbus module written in Ada (Matra Space)
- the EGNOS air navigation system (Alcatel Space), whose specifications are being finalized, will be written in Ada (Aonix' SMART)

François Gody, Matra Marconi Space, Toulouse, France

From: David Luc <Luc.David@ganil.fr>
Date: Fri, 14 Sep 2001 09:42:29 +0200
Subject: Ada en entreprise
URL: <ada-france>/2001-September/000392.html

The control system of the "Grand Accelerateur National d'Ions Lourds", situated in Caen, Calvados. All our programming is done in Ada, the real-time parts as well as the human-machine interfaces. We develop all our code ourselves, as we don't have the budget to subcontract software. [...]

Luc David, Groupe Informatique Machine, G.A.N.I.L., Caen, France

From: Lionel Draghi <Lionel.Draghi@free.fr>
Date: Sat, 15 Sep 2001 22:08:30 GMT
Subject: [Emploi] SII Toulouse
Newsgroups: fr.comp.lang.ada

[Extracts translated from French: -- dc]

SII Toulouse [...] specialised in industrial software [...] is looking for engineers embedded software with Ada experience, if possible in a real-time environment (VRTX, VxWorks, Tornado, LynxOS); and an experienced project leader embedded real-time software, familiar with the following languages and tools: Ada, Assembler, C, C++, VxWorks, Tornado, VRTX, Attol. [...]

From: Lionel Draghi <Lionel.Draghi@free.fr>
Date: Sat, 15 Sep 2001 22:16:07 GMT
Subject: [Emploi] Quaternove
Newsgroups: fr.comp.lang.ada

[Extracts translated from French: -- dc]

Ada development engineers, Paris region, defense & aerospace, from beginner up to 5 years experience. Quaternove, [...] specialised in industrial software, optical and electronic engineering. [...] competences in Ada 83 and/or Ada 95 in a Unix environment. [...]

Date: Thu, 20 Sep 2001 09:22:37 +0200
Subject: ada-france: Emplois & stages
URL: http://www.ada-france.org/EMPLOIS/offres.html

[At URL http://www.ada-france.org/EMPLOIS Ada-France maintains a web-page with Ada-related job offers from its members, and currently has entries from: -- dc]

- Rational France, technical consultants
- Aonix, software development engineers
- CS Aerospace, project leaders, engineers, embedded real-time systems
- Adalog, Ada developers

From: Claude Marinier <claude.marinier@dreo.dnd.ca>
Date: Wed, 26 Sep 2001 09:44:41 -0400
Subject: Re: gnat and heap size
Newsgroups: comp.lang.ada

We want to use large arrays (well, large for us: 10000 x 10000 complex numbers). We are using GNAT 3.13p on Solaris 7. [...] The application is an electromagnetic simulation. [...]

Claude Marinier, Information Technology Group, Defence Research Establishment Ottawa (DREO), 3701 Carling Avenue, Ottawa, Ontario, K1A 0Z4, Canada, http://www.dreo.dnd.ca

From: "PlanetRecruit.com" <mailout@planetrecruit.com>
Date: Thu, 04 Oct 2001 07:05:36 +0000
*Subject: ** 3 NEW Jobs from PlanetRecruit.com (04/10/2001) **

Ada Software specialist (Belgium)

You must have a minimum of 2 years Ada experience, experience of Development of Safety Critical software and Rational Apex as a user.

Functional validation Engineer (Belgium)

Minimum of 2 years experience in Ada on a single project. Experience in the development of safety critical software, the development of real-time and infrastructure software. Tools required Rational Apex (as user), other Ada Compilers (Aonix &/or Greenhills) Unix and Integration tools (eg logic analyser). Plus Testing experience at requirement and module level. Need to be French speaking.

Functional Validation Engineer (Belgium)

I have 2 contract positions for a [...] functional validation engineer with 3-5 years current experience with Ada, development of safety critical software, Rational Apex (as user), (Aonix &/or Greenhills), Unix, Integration tools and Knowledge of Ada run-time, Power Pc target and understanding of real-time control systems and testing experience at requirement [...]

From: "PlanetRecruit.com" <mailout@planetrecruit.com>
Date: Sat, 20 Oct 2001 06:20:41 +0000
*Subject: ** 1 NEW Jobs from PlanetRecruit.com (20/10/2001) **

Software Functional Validation Engineers (Belgium)

Looking for Validation/Integration Engineers to work in Belgium for 6 months. You must have min 4 yrs exp of Ada83 or Ada95, ideally 2 years or more on 1 project. Other skills req: Yourdon, HOOD, SSADM, UML, Teamwork, Rational Apex, Ipsys, Artisan, Word, Excel, PVCS, SCCS, Apex, Clearcase, AdaTest, LDRA Testbed, TeamTest, ATTOL, Change Control, SPARC Ada. You will be involved in full project life cycle. French language will be a great benefit.

From: JPV <jean-philippe_vassilakis@jyhoriba.fr>
Date: Tue, 23 Oct 2001 10:36:48 +0200
Subject: Re: Blending Delphi with Ada95
To: team-ada@acm.org

[...] this is what we do on Windows projects for many years. Ada95 for the business logic. Delphi (sometimes VC++) for the user interface. Middleware by home made messaging allowing a LAN between those 2 parts. Works really good.

Jean-Philippe Vassilakis, Jobin-Yvon Thin Films Division, Software for Process Control & Ellipsometers, 5 Avenue Arago, 91380 Chilly-Mazarin, France

From: "PlanetRecruit.com" <mailout@planetrecruit.com>
Date: Tue, 13 Nov 2001 07:21:25 +0000
*Subject: ** 3 NEW Jobs from PlanetRecruit.com (13/11/2001) **

Ada Software Engineer (Belgium)

World Class International Company requires an Ada Software Engineer to intergrate into a project involving the design, development, testing and writing of documentation. You will play an integral part in the development team, and ideally you have at least 2 years commercial Ada experience with a Computer Science/Engineering degree.

Ada Software Engineer (Belgium)

My client, the leader in its field, urgently requires an experienced Ada Engineer to join cutting edge development team working on various projects. You will have at least 2 years commercial experience using Ada 83-95 with knowledge of C++ a distinct advantage. You will also be of graduate calibre with a degree in Computer Sciences/Engineering (Civil/Industrial). [...]

From: "PlanetRecruit.com" <mailout@planetrecruit.com>
Date: Wed, 14 Nov 2001 06:13:19 +0000
*Subject: ** 1 NEW Jobs from PlanetRecruit.com (14/11/2001) **

Ada Software Engineer (Belgium)

Internationally renowned market leader requires an experienced Ada Software Engineer to join expanding development team. You will play an integral part in the team, and ideally you have at least 2 years commercial Ada experience with a Computer Science/Engineering degree. Duties include the design, developing, implementing and testing of documentation for various projects.

Date: Tue, 20 Nov 2001 08:42:13 +0100 (MET)
Subject: Ada-related job announcements

[Posted on the monster.be website. -- dc]

Belgium, Brussels, Ada Developer Required

Required for an urgent contract in Brussels. You should have good experience of Ada and knowledge of a scripting language like Perl. You will be working on a large project performing code review in Ada and coaching the team. French speaker preferred. [...]

From: "PlanetRecruit.com" <mailout@planetrecruit.com>
Date: Wed, 21 Nov 2001 06:54:33 +0000
*Subject: ** 1 NEW Jobs from PlanetRecruit.com (21/11/2001) **

Ada Engineers/Project Managers (Belgium)

My client requires French or Dutch speaking Ada Engineers to work on important projects. Technically you will have good commercial experience of Ada, C++, OOAD (UML) and good knowledge of Unix and NT. Successful candidates will also have Degree in Software Engineering (or related technical discipline) and excellent communication skills.

From: "PlanetRecruit.com" <mailout@planetrecruit.com>
Date: Thu, 22 Nov 2001 06:40:05 +0000
*Subject: ** 3 NEW Jobs from PlanetRecruit.com (22/11/2001) **

[...] Ada Software Engineer (Belgium)

Leading Consultancy requires experienced engineers and project managers to work on client projects. You must have good commercial experience of Ada design and development, with knowledge of real-time/embedded systems a distinct advantage. French/Dutch speaking a bonus. [...]

From: Erik Wischmann <erik.wischmann@euro-telematik.de>
Date: Thu, 22 Nov 2001 12:14:15 +0100
Organization: Euro Telematik AG
Subject: Job Offer: Ada SW Engineers
Newsgroups: comp.lang.ada

Ada SW-Engineers needed!

Euro Telematik AG was founded in 1998 as a Management Buy-Out of then Daimler-Benz Aerospace (DASA). We are offering products and engineering services in the aerospace and road telematics market.

For our ongoing expansion of our engineering business in the aerospace sector, we are currently seeking software engineers with a good knowledge of Ada. Avionics background is preferred, but not a necessity.

The offer is for a long-term employment. Since our customer is working on military projects, all employees in this sector will be subject to clearance screening.

Work will be performed at our customer's facilities in Ottobrunn near Munich. The job will include: SW-Design, Coding, Module- and Sub-System testing, and documentation on enhanced avionic systems for military aircraft. [...]

[And from another message: -- dc]

We are also looking for a SW engineer to work in Edinburgh, Scotland. The other mentioned conditions apply to this job, as well. [...]

From: "PlanetRecruit.com" <mailout@planetrecruit.com>
Date: Fri, 23 Nov 2001 06:04:08 +0000
*Subject: ** 4 NEW Jobs from PlanetRecruit.com (23/11/2001) **

Ada Project Managers (Belgium)

Ada Project Managers are urgently required by market leading consultancy to work on full life cycle projects for aviation industry. You will have knowledge of Ada design and development along with Real-Time constrained methodologies. Fluency in either Dutch or French a massive advantage. Opportunities for perm roles as well.

Ada Software Engineer (Belgium)

My client urgently requires experienced Ada Engineers to work in their embedded competence centre for the aeronautic and railway sectors. You will have excellent Ada Development and Design experience, with a knowledge of real-time methodologies a distinct advantage. [...]

The Future of Ada

From: dewar@gnat.com (Robert Dewar)
Date: 8 Oct 2001 08:02:52 -0700
Subject: Re: is Ada dying?
Newsgroups: comp.lang.ada

[In a thread spreading a lot of FUD, such as "it seems Ada is dying", "from what I can see very few people use Ada", "Ada's parents (the U.S. DoD) are dropping Ada 95", etc. -- dc]

Isn't it odd in this field that if a technology is not dominant (by number of applications), then it is considered dead (examples, Pascal, PL/1, OS/2 ... all of which are alive and used for many important applications). The trade press has even announced that Java is dead on the client side, and no doubt given

Microsoft's decision to exclude Java from XP, will pronounce it completely dead. Sometimes people even decide that widely used technologies are dead. I once heard a high up official in the DoD tell me that no one outside the DoD used COBOL any more (that statement was made over a year ago!)

The U.S. DoD is not "dropping Ada", to think this is as wrong as to think that everyone in the DoD was using Ada during the mandate. The actual fact is that, not at all surprisingly, some people in the DoD like Ada, and fight to do as much as possible in Ada, and some people in the DoD dislike it, and fight to do as much as possible in some other language (C++ or even Java).

Ada is certainly not dead, and use of Ada will continue for a long time. Will usage increase or decrease? Hard to say. Here at Ada Core Technologies, we see a steady increase in use. This can of course be due to three factors

a) people updating from Ada 83 to Ada 95;

b) people shifting from other Ada technologies to GNAT;

c) new projects being started in Ada.

We certainly know some projects that are in category c, but it is hard to know what the division between these three is. In any case, regardless of what other vendors do, ACT expects to be supporting Ada for a long time to come, and to continue to do active development and enhancements to the GNAT technology (we already have a long list of enhancements that have been made for version 3.15). As you know from our web site, 3.14 also had a long list of enhancements (and we expect to see 3.14 public versions out soon for selected targets).

If you want to learn a dominant technology that is very widely used, I would suggest Visual Basic or COBOL, there is a big demand for people in both areas, and these are still among the most widely used languages. But if you want to learn Ada, you will find that

a) You acquire skills and knowledge that are useful not only in Ada, but in other arenas.

b) There are definitely jobs for competent Ada programmers.

Robert Dewar, Ada Core Technologies

Ada in Context

On Separation of Interface and Implementation

From: chris.m.moore@amsjv.com (Chris M. Moore)
Date: Wed, 05 Sep 2001 11:08:37 GMT
Organization: Alenia Marconi Systems
Subject: Another good URL for all you C++ haters
Newsgroups: comp.lang.ada

[Excerpt from] http://www.byte.com/documents/s=1271/byt20010831s0001/0903_nichols.html

Even Floyd, who loves the language, admits that, "Its greatest weaknesses is a flaw in the interface/implementation separation. A class exposes its private data/methods to the world and a programmer must use some tricks (that should be supported directly in the language) to overcome this."

Chris M. Moore, Software engineer

From: Tucker Taft <stt@avercom.net>
Date: Fri, 07 Sep 2001 13:54:52 -0400
Organization: AverStar (formerly Intermetrics) Burlington, MA USA
Subject: Re: Another good URL for all you C++ haters
Newsgroups: comp.lang.ada

> Doesn't have Ada a similar problem?

Ada requires the full definition of a private type to be in the package spec, but it doesn't require the declaration of "helper" functions the way C++ does. Helper functions can be declared in the package body, without perturbing the spec.

> In C++, one can use a "friend" class to contain the helper functions, but that is pretty obscure.

Note that both C++ and Ada allow access/pointer types to be declared without exposing the full details of the target type.

Tucker Taft, stt@avercom.net, http://www.avercom.net, Chief Technology Officer, AverCom Corporation (A Titan Company), Bedford, MA, USA

From: James Rogers <jimmaureenrogers@worldnet.att.net>
Date: Wed, 05 Sep 2001 14:43:13 GMT
Subject: Re: Another good URL for all you C++ haters
Newsgroups: comp.lang.ada

The author of this article claims he spoke to a lot of C++ programmers. He may have. He also posted a questionnaire on comp.lang.c++.moderated asking specific questions about C++. The thread in that newsgroup is titled "Whether C++?". I know, it should be "Whither", as in the article, but he misspelled the title in his posting.

I learned a lot from the responses to this article. Mostly I learned that many C++ programmers know only C, C++, Java, and Perl. They do not know about any languages not descended from C syntax.

For instance, several responders stated that the primary strength of C++ is that it is the only language that gives you a choice of design paradigms (OO and non-OO) as well as generics and low level programming capability. This is a clear indication that those responders have no understanding of Ada.

There was general agreement that C++ is a very complex language. It is so complex that, four years after its standardization, there are no compilers fully compliant with the standard. Some people see this as a problem. Some people see it as a good thing. I only hope I do not have to work with anybody who thinks it is best to use a tool that takes years of study before it can be used correctly.

I do not hate C++. I do have an aversion to ignorance masquerading as knowledge. My conclusion from the thread was that many people prefer C++ because they are ignorant of any alternatives.

Jim Rogers, Colorado Springs, Colorado USA

From: James Rogers <jimmaureenrogers@worldnet.att.net>
Date: Wed, 05 Sep 2001 21:49:24 GMT
Subject: Re: Another good URL for all you C++ haters
Newsgroups: comp.lang.ada

> Doesn't have Ada a similar problem? [see above -- dc]

Similar but not quite as severe.

Ada private data and subprograms are what C++ calls "protected" data/methods. Ada's equivalent to C++ private data/methods is data and methods defined in a package body but not in the specification. C++ public, private, and protected data and functions are normally declared in a header file. Functions are then defined in the .CPP (source) file.

There is no automatic enforcement of separation of implementation and interface in C++. The entire implementation can be placed in the C++ header file. As with so many other C++ features, only good practice can produce good separation of interface and implementation.

Compare both C++ and Ada to Java. Most Java classes are defined without benefit of an interface. In fact, Java interfaces are most typically used to define a call-back interface rather than to define common classes. Java does not even provide you the choice of using a header file. Java fans proclaim a lack of separation of interface and implementation to be an advantage because you have less code duplication.

From: James Rogers <jimmaureenrogers@worldnet.att.net>
Date: Thu, 06 Sep 2001 16:57:52 GMT
Subject: Re: Another good URL for all you C++ haters
Newsgroups: comp.lang.ada

[On the remark that "Java fans proclaim a lack of separation of interface and

implementation to be an advantage because you have less code duplication." -- dc]

> In my experience, Java fans claim that the Interface/Implementation separation feature of OO is provided by the Java Interface structure and Class structure (i.e. the Interface provides the Interface, and the Class provides the Implementation). This seems a very high level viewpoint corresponding to something like a pure virtual class (C++), or abstract tagged type (Ada) rather than what Ada programmers often see as the separation, i.e. the package spec/body. Personally I believe it is a combination of both.

Java also allows the definition of abstract classes. They have the expected restrictions such as not being able to create an instance of an abstract class. Java interfaces are treated differently by the language. In Java you can extend (inherit from) only one class, producing single inheritance similar to Ada's. On the other hand, you can implement any number of interfaces, and interfaces may inherit from other interfaces.

In many cases you can program Java entirely without the use of interfaces. Exceptions to this include the use of event handlers. Java event handlers are commonly used in GUIs and the reusable code modules called Java Beans.

Note that, for Java Beans, only the event handler interface is required. There is no interface requirement for the get and set methods for Bean properties. Common practice, then, provides only partial separation of interface and implementation for Java Beans. Builder tools that use Java Beans employ introspection to evaluate Bean properties. In this case the implementation must report its interface at run time.

Furthermore, there is no way to specify a constructor in a Java interface. Thus, even if you are diligent about building your Java classes from interfaces, you must still omit critical parts of the class definition.

Why is introspection not as good as explicit separation of interface and implementation? Not every program uses introspection. Furthermore, introspection is a run-time activity. This means that compilers must have access to full implementation code, even though Java exclusively uses a shared library (aka dll) model for linking. There is no way in Java to perform an early compilation of just the interfaces, as can be done in Ada. Lack of separation inhibits large team development.

On Languages, Productivity and Quality

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Mon, 1 Oct 2001 11:02:45 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

> I come from the M$ world. I make my living as a programmer and as such I don't have to tell you how aggravating it gets some times. Anyway, to make a long story short, I have a project in mind that needs a rock solid OS. It can't crash or freeze like windblows does cause it's aircraft instrumentation. [...]

I've used Ada to build jet and rocket engine controls with no COTS OS (bare machine - our own "executive") where failure was not an option. Compared to other languages we used, we got a doubling of productivity and a four-fold reduction in errors. We considered it a major step forward in reliability & could see no reason to go elsewhere.

I've not used it, but I believe RTEMS was a suitable OS for military missile technology and was written in Ada. I believe it has been used in mission critical projects, so it has a track record. Its definitely worth looking into for a serious avionics project. [See "OAR - RTEMS Operating System" in this AUJ issue. -- dc]

Marin David Condic, Senior Software Engineer, Pace Micro Technology Americas, www.pacemicro.com, http://www.mcondic.com/

From: "Pat Rogers" <progers@classwide.com>
Date: Mon, 01 Oct 2001 17:54:35 GMT
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

[Someone who was involved in a "good" C project and a "bad" Ada project responded: -- dc]

> Good coders can write good code regardless of language. Language won't make marginal coders, marginal coding teams, or bad design any better. You missed his point. The question is not whether language is a replacement for talent, and he did not assert that it was impossible to make high reliability code with anything other than Ada. The question is "for how much money?". His point (if I may be so bold) is that they did it much more economically in Ada. Others have seen these results too.

Patrick Rogers, http://www.classwide.com

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Mon, 1 Oct 2001 14:42:36 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel,comp.realtime,comp.lang.ada

> [...] Good coders can write good code regardless of language. [...]

I currently am building embedded systems in C so it isn't as if I have no experience on the other side. I would agree that good software engineers can produce good code in any language - including assembler. But after having acquired years of experience with Ada, I believe that the job is a *lot* easier using that language as opposed to C/C++.

I would caution against relying on what I like to call the "Any *Competent* Programmer" argument. All of us on any given day make stupid little mistakes and I've had metric data demonstrating that a team of highly "competent" realtime software engineers with years of experience in engine controls make 4 times fewer mistakes that made it into the lab using Ada as compared to other languages. These were engineers that for the most part all had at least 10 years of experience in realtime systems and working with engine controls specifically. They were not greenhorns or morons. Yet their productivity doubled and their error rates were reduced to a fourth of what they were before. Many were skeptical of Ada initially but grew to appreciate the language as they learned to use it. Languages can and do make measurable differences in the quality & cost of the end product.

Note that the engine controls we made prior to that still had to work with extremely high reliability, so I won't dispute that you can build a solid product in other languages. It just costs more and requires more time. Now that I'm working on digital TV equipment in C, I am once again reminded of this fact as I have to constantly track down and fix errors that would otherwise be caught automatically by a more secure programming language. The box still needs to be reliable, so we'll end up spending lots of the stockholder's money testing and fixing it, rather than making more and better products - but there isn't anything I can do about that since too many of my associates are entrenched in C and too much of the infrastructure is reliant on C. The data is there to demonstrate that a better, more productive job can be done in other languages - specifically Ada - but it is hard to beat the entrenched establishment that is stuck inside the box we're all supposed to think outside of. Oh well...

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Mon, 1 Oct 2001 14:52:01 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

[In response to Pat Rogers' message quoted above: -- dc]

[...] That is exactly my point. Productivity doubled and errors were reduced by a factor of four. (The latter probably having a significant impact on the former.) That has to translate into lower cost and higher reliability. You can get good systems in any language - you just might have to spend the next 5 years debugging & patching in the lab to get there. I'd rather get all the automated help I can find by way of Ada's checking, structural integrity and organizational capabilities. That will preserve the stockholder's money for something more productive.

Also, I would contend that there are likely to be other languages that show similar improvements over C/C++ in terms of productivity and error rates. I just don't have any data on other languages that provide similar safety capabilities. [...]

From: dmitry@elros.cbb-automation.de (Dmitry Kazakov)
Date: Tue, 02 Oct 2001 08:06:44 GMT
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

[In reply to "You can get good systems in any language - you just might have to spend the next 5 years debugging & patching in the lab to get there" someone wrote: -- dc]

> Cool. More job security! The boss can't fire me, because the product still has bugs in it that needs to be fixed (and of course no one else can fix them because no one else can figure the code out because we did not write any design documents).

[and someone else: -- dc]

> But the customers can fire the whole company!

Yes they can but will not (:-)). When you develop in Ada and thus have a higher quality product, the customer has an ability to concentrate on real issues (and discover that the whole project makes no sense at all (:-)). With C++ (powered by Windows) he concentrates on bug reports, updates, service packs etc, which gives him a warm feeling that "the process goes" and that he pays for a damn good and hard work. It sounds cynical, but sometimes the worse is better...

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Tue, 2 Oct 2001 13:43:15 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

[In reply to a request for published information to illustrate or support the statement "Compared to other languages we used, we got a doubling of productivity and a four-fold reduction in errors.": -- dc]

Unfortunately, the study was an internal one for a company that I am no longer employed by and the results were never published. Companies tend to get a little reluctant sometimes when it comes to publicizing things about their internal operations - especially if they perceive it to have some competitive advantage. [See "Measuring Productivity" in AUJ 22.3 (September 2001), p.167. -- dc]

However, this was not the only source for productivity data. A better source for a study can be found at: http://www.stsc.hill.af.mil/crosstalk/2000/aug/mccormick.asp and http://www.tcsigada.org/meeting/feb99mtg.htm and http://www.rational.com/products/whitepapers/337.jsp

Dr McCormick's studies come about as close to a controlled experiment in software productivity as I've seen. Other industry studies are plagued by the fact that you seldom build the same thing twice. His experience is based on multiple groups building software to satisfy identical requirements under nearly identical circumstances. That, I think, offers a little more weight to his results than might be given to my own internal study. [See "Software Engineering: On the Right Track" in this AUJ issue. -- dc] But in either case, the conclusion was that use of Ada bought an improvement in productivity and a reduction in errors. Of course, YMMV. :-)

From: Kilgallen@SpamCop.net (Larry Kilgallen)
Date: 2 Oct 2001 13:40:06 -0500
Organization: LJK Software
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

[On "they did it much more economically in Ada. Others have seen these results too.": -- dc]

> Man hours, maybe, if you have a team that already has significant Ada experience. Include costs of compiler, language & tool training, development & emulation platforms, & future maintenance plus factor in employee turnover, may reveal a different economic interpretation.

Absolutely! Ada is much easier to learn to a satisfactory level than C++, and also suffers less from the phenomena of people who used it a little in a class passing themselves off as expert.

[Others responded as well, e.g. on compiler cost: -- dc]

Comparable, ranging from free to pricey, as is the case for any language. Surely you're not thinking of the 1980's, are you? Ancient history. [And:] Yes, sure. With programmers costing $50/hr and sometimes double that, I can see that a $200 or so for a compiler is really too much to pay for.

[On cost of language & tool training:] Comparable to any other language. [And:] If you know a programmer who needs more than 2 weeks training to pick up a new language, let me know, so I make sure I do not hire them.

[On future maintenance:] A big win for Ada, compared to other languages. [And:] Yes, good point. Languages such as C and Perl are the best language for future maintenance, they provide years and years of sustained maintenance for us programmers to live on.

[On employee turnover:] Yup. The Ada people tend to stay longer. That's just my personal observation. [And:] Those dim a dozzen programmers, we do not want to lose them, those who need years to learn a new language, and get confused when they see BEGIN instead of "{". We got to keep those ones.

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Tue, 2 Oct 2001 14:44:12 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

Q: What do you get when you add one plus one?

Mathematician: 2.0

Engineer: 2.0 +/- 0.001

Accountant: What do you want it to be? :-)

The point: If you *want* the answer to be "Program everything in C", you can probably find a way to stack up a bunch of reasons why it is that "C" is the right answer. OTOH, if you truly have an open mind and want to investigate what might be an optimal solution for development of a particluar kind of system, you might look at some of the studies I cited elsewhere and the experience attested to by a variety of users and *possibly* come to the conclusion that it is worth a shot at using Ada (or other languages that provide more safety) and then think that the one-time transitional overhead is worth it.

Remember that at one time Java didn't exist, yet a bunch of people didn't see any big problem in adopting it for a variety of applications. It required training, investment in compilers, tools, etc. as you pointed out. But they believed they were going to get something as a result. Not really different for Ada, would you think?

BTW: As for cost of compilers and tools, depending on your platform the cost can be as little as $0.00. See: http://www.adapower.org/ for links to free compilers, free GUI builders, free bindings, free etc.

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Wed, 3 Oct 2001 13:39:14 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

> [...] if there *were* numbers on C vs. Fortran, I wouldn't be shocked to see Fortran (even F77) come out on top. C's overeliance on pointers cause a great deal of its problems. But this is all theoretical. The numbers on Ada and C actually exist.

If anecdotes count for anything, I could enumerate a long list of hours spent in the last few weeks (We're busy trying to test quality into the system! :-) where I've been tracking down memory leaks, dangling pointers, illegal memory references, and bad array indexes all because of pointer usage and/or lack of checks in C which - FWIW - wouldn't happen in Ada because of the a) limited use of pointers, b) scope rules, c) compile time checks or d) run time checks.

I recall a Bell Labs study of bugs in one of their big switching applications in which the report found a whole slew of common bugs which were classified into types & examples given. Their recommendation was to institute coding standards and code reviews to check for those specific problems. The overwhelming bulk of the bug types they identified were impossible to commit in Ada because of compile or runtime checks and the rest were highly unlikely because Ada doesn't depend on addresses/pointers for everything in sight. [See "Ada is More Productive" in AUJ 20.4 (January 2000), pp.262-263. -- dc]

It's anectdotal, but it adds a level of experience and reasoning that explains *why* the studies indicate superior productivity/error rates with Ada vs C.

On Languages, Reliability and Time-To-Market

From: Ted Dennison <dennison@telepath.com>
Date: Wed, 03 Oct 2001 17:31:30 GMT
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

> [...] If your job ads read "I need Microsoft Visual C++ / IBM-PC / Windows95 programmers..." what will you do with them if Linux becomes the hot craze? Or if the architecture changes to Sun/Unix? Or (God Forbid!) some *other* language becomes the hot item this week? Makes more sense to me to find adaptable engineers, 'cause the world of computers is *always* changing.

That's one of the reasons I think Ada is a good choice. If you do your software right, it is probably going to end up lasting for many years. Thus if you choose a language that has *nothing* to offer other than it present-day market share or "coolness", you might as well choose the one with the prettiest box, for all the good it will have do you 5 years down the line.

Ada isn't about market share or slick marketing or cool logos. Ada is about writing fast, bullet-proof software, period. Those are timeless values that will *never* quit paying you dividends, no matter what language is cool 3 years(or weeks) from now.

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Wed, 3 Oct 2001 13:52:09 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

Well, there *are* problem domains in which 5 years represents 10 generations of product. :-) You and I may have lots of experience with long-lived systems and see the value of Ada there rather readily. For faster product cycles, things like maintenance may be less of a concern. But then, so would be the concern about finding programmers who know the language 5 years from now.

In fast product cycle environments, I'd think the advantages of high reliability are still of major importance. Who wants to put out an electronic gadget and sell millions of them only to discover some fatal flaw after they ship that forces recalls and/or lawsuits? We all know it has happened and could probably cite lots of cases here. Ada wouldn't guarantee it doesn't happen, but it helps minimize that risk.

From: Ted Dennison <dennison@telepath.com>
Date: Wed, 03 Oct 2001 20:07:43 GMT
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

I wasn't talking specificaly about maintenance. Just because you have to *develop* quickly doesn't mean that people won't be trying to use that software 5 years from now. It also doesn't mean that developers won't be trying to use your sources 5 years from now. In a short-cycle environment reuse of old code from the previous generation is vital to a project's success. What you don't want in that enviroment is to find yourself stuck with a language that was designed to promote *itself*, rather than reuse.

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Wed, 3 Oct 2001 16:27:11 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

I was rather thinking about the case where someone develops software for a product and the product lives for a relatively short time and the *next* product that comes along reuses little to none of the existing code because it is sufficiently "new" to warrant a whole new development. You might see this in some kinds of consumer electronics products and some PC types of apps, where basically a whole new look & feel needs to be developed every year to 18 months. In effect, the developers are building throw-away code.

Granted, part of the reason this may be done is that the language of implementation makes it sufficiently hard to maintain, enhance or reuse, so it becomes more cost effective to pitch it and start over. Some of the reason it might get done is simply to guarantee a difference with every release. Some of the reason may be because the company wants to avoid the costs involved in building systems that will hang around for a long time or have big reuse factors. [...]

I'll easily concede that Ada buys you a lot for long-lived systems or developing reusable code or any of the conditions that may keep what you build around for five years. I'd just offer that even when you don't have long-lifespan concerns, Ada can make a lot of sense from a reliability and time-to-market perspective as well.

From: Preben Randhol <randhol+abuse@pvv.org>
Date: Wed, 3 Oct 2001 23:48:47 +0000 (UTC)
Organization: Norwegian university of science and technology
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

> [...] Everything you say is true, but when talking to people who develop throw-away code, the emphasis should be on time-to-market and reliability rather than long-term benefits such as reduced maintenance or reusable code.

I cannot see that Ada wouldn't accommodate on all these areas. :-)

> Otherwise, folks in a fast-moving throw-away market may find the argument interesting, but not compelling.

My hope is that the folks in the fast-moving throw-away market are throwing themselves out with the bath water :-) I mean "Good Enough" isn't "Good Enough" for the consumer when he suddenly finds himself with a negative bank account on Monday morning due to a software error in a banks software which is fixed soonest next day more likely next week.

I don't expect the Software Industry to clean up its act themselves. I'm hoping the consumers take action and start realizing that it _may_ be cheaper in the long run to pay for good quality and avoid all costy troubles in the future that cheap JIT code gives. It is funny as I just see now that Microsoft is saying that they will "Step Up Software Security" (http://linuxtoday.com/news_story.php3?ltsn=2001-10-03-019-20-SC-MS). I guess the only reason for this statement is recent suggestions from Gartner Group (and others) that consumers should replace Windows with other OSes due to poor security.

From: pete@nospam <pete_member@newsguy.com>
Date: 3 Oct 2001 16:56:25 -0700
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

> [...] when talking to people who develop throw-away code, the emphasis should be on time-to-market and reliability [...]

But, but, but, code reuse is what will make time-to-market much smaller. Ada is one of the best languages for designing code reuse, which means if you want very short time-to-market, then Ada is one of the best choices.

> Otherwise, folks in a fast-moving throw-away market may find the argument interesting, but not compelling.

Until they see that code reuse = profit and shorter time-to-market. Of course, writing packages with the idea of reuse requires more time and effort, but it will soon pay of (in the next project).

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Thu, 4 Oct 2001 09:51:16 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: linux.dev.kernel, comp.realtime, comp.lang.ada

Never said I was against code reuse. [...] I said that in some types of development for some problem domains, code reuse is not very interesting or important - for a variety of reasons. Of course reuse is a good way to speed time to market. Where would we be without APIs to operating systems or GUI interfaces, etc? That's "reuse" and the more of it you can do, the better.

The point is, if a given application domain isn't interested in reuse, then there are still *other* reasons why Ada is an advantage. Those areas should be emphasized when talking to developers who don't have an interest in the long term benefits Ada has.

From: "David Botton" <David@Botton.com>
Date: Thu, 4 Oct 2001 13:19:03 -0400
Subject: Re: Is Linux right for Embedded?
Newsgroups: comp.lang.ada

> And in some cases, missing that first time window of delivery means that even if you did built a fantasically designed and reusable system, the market isn't going to afford you the opportunity to make use of it since you won't be around for that second release :-)

Ada also excels here since many "human" errors that quickly start to be a drain on productivity are found at compile time. Being both a C++ and Ada programmer, I find that Ada is exceptionally well suited to getting things done quickly the first time and on time. I even often prototype in Ada before writing C++ (since C++ is often "required"...).

Strange Criteria for Language Comparisons

From: minyard@acm.org (Corey Minyard)
Date: Tue, 16 Oct 2001 15:30:53 GMT
Subject: Re: "Size" of Ada vs. C++
Newsgroups: comp.lang.ada

[From a thread comparing the "size" of programming languages based on the number of pages of the reference manual. -- dc]

> In any case, I doubt that the "size" of the language tells us much about the language itself. Perhaps there are additional factors that indicate that the language is too big (for example, if no complete implementations exist), but size is not a problem per se. For example, I think most if not all Ada programmers appreciate the elaboration semantics. They are rather complex, but without them, very annoying problems would arise.

I'm not arguing that point, the "size" of a language is quite meaningless, otherwise we would all be programming Turing machines :-). Someone was saying that Ada might not be smaller than C++, and I was giving some fuel for the debate.

But, it's quite amazing to me that Ada can include all the tasking semantics, a full set of I/O libaries, all the fancy numeric types, annexes for real-time, systems programming, information systems, distributed systems, and safety and security, and still weigh in with a smaller and more usable specification than C++ and a simpler syntax.

From: Richard Riehle <richard@adaworks.com>
Date: Tue, 16 Oct 2001 22:39:55 -0700
Organization: AdaWorks Software Engineering
Subject: Re: "Size" of Ada vs. C++
Newsgroups: comp.lang.ada

> Anyway, I think it's rather strange to judge a language by the "size" of its definition.

Absolutely correct. I have seen people compare languages based on the number of reserved words (less is better), the size of a "hello world" program's executable (less is better), the number of predefined libraries (more is better), the number of people who use it for programming (more is better), the number of ads in the help wanted section (more is better), and on and on and on. There is no end to the silliness that people use to compare programming languages. The Modula-3 exercise that attempts to define a language in some number of pages of the reference manual also approaches absurdity.

A programming language needs to be judged on its expressiveness first. One factor in this judgement is, how well it expresses the software problems it is intended to solve. We could reframe this as, how well does the solution space (represented by the language and its corresponding tools) map to the problem space? Another factor derives from Donald Knuth's notion of "Literate Programming." This notion suggests the question, how well does the language support the human process of developing software? There are other factors, but this margin is too small to enumerated them all.

In evaluating a programming language, we need to define qualitative criteria as well as quantitative criteria. Sadly, most of quantitative criteria are flawed due to the immature reasoning that leads to their selection. Sadder still, such reasoning is likely to prevail for a very long time, given the current state of software engineering practice. Sorry for my pessimism.

Why Do Some Projects Move to C++?

From: Ted Dennison <dennison@telepath.com>
Date: Thu, 01 Nov 2001 14:55:03 GMT
Subject: Re: Joint Strike Fighter
Newsgroups: comp.lang.ada

> [...] many younger employees want C++ because that is what they learned in school (in many cases) so that's their skill set, or, yes, they want to get paid to learn it in the first place, for job security and mobility.

[...] The folks pushing for C++ (or other languages for that matter) over Ada are almost never the experienced engineers who will actually have to *use* the damn language. Some fresh out of college types would prefer not to use [Ada] too, but they generally come around after they actually use it for a while.

Way back in the "mandate" days I worked on one project where our lead software developer basicly took over the entire project (with the backing of the rest

of us developers of course). Program management had our Ada waiver all ready, and was absolutly shocked when we didn't want it. :-) That was the best-run project it has ever been my privelege to work on, btw.

From: Richard Riehle <richard@adaworks.com>
Date: Sat, 03 Nov 2001 08:58:03 -0800
Organization: AdaWorks Software Engineering
Subject: Re: Joint Strike Fighter
Newsgroups: comp.lang.ada

> A lot of things go into the decision to move to C++. The techies may want it because it is what they know or what they want to know for career enhancement. Management may favor it because they figure its easier to hire people who know C++, it has the appearance of where the computer industry is going and "Nobody ever got fired for buying C++..." lemming mentality.

One of the most important benefits of C++ is that, once people have enough experience with it, it becomes obvious how inherently hideous it is. For those with a cursory knowledge of C++, or those whose experience with it is shallow, the language can look quite appealing. Only after wrestling with some of the more entertaining aspects of the language in the production of large-scale software does one begin to realize that it falls far short of what they enjoyed with Ada. I know former Ada software developers who are now engaged, by managment fiat, in using C++. They were at first full of enthusiasm for moving to a better resume-building language. They looked at the simple C++ class model and considered it easier to understand than Ada's package model. In those early stages, C++ seemed more accessible.

Ahhhh, but "the devil is in the details." The more preceptive among them eventually discover that whoever said, "C++ is its own virus" was on to something. Unfortunately, once they have committed to a particular course of action, they are stuck with it. It is too late for "buyer's remorse" to save them. I believe it was Thorsten Veblen who used the phrase, "Caveat Emptor," to describe this situation. It makes one wonder whether those who are willing to risk using C++ for a safety-critical project are "playing dice with the universe."

From: Jeffrey Carter <jrcarter@acm.org>
Date: Sat, 03 Nov 2001 18:52:21 GMT
Subject: Re: Joint Strike Fighter
Newsgroups: comp.lang.ada

> One of the most important benefits of C++ is that, once people have enough experience with it, it becomes obvious how inherently hideous it is.

That's an interesting use of "benefit". :)

There is a column in ESP called "Programmer's Toolbox". The author used to present algorithms in Turbo Pascal; while I thought an ALGOL based pseudocode might have been a better choice, the results were generally easily understood, and I found the column interesting and sometimes useful. Then he changed to the then-current fad language, C++. I found the results difficult to understand. When he devoted 3 consecutive columns to the intricacies of redefining "=" (assignment), I stopped reading the column. That was also when I stopped thinking that C++ might have some redeeming social qualities. I think he eventually gave up C++ and is now using C, but with no improvement in clarity.

As to language choice for large US defense contracts, in 26 years of professional software development, I have seen numerous such projects, and am acquainted with more through colleagues. They almost all seem to suffer from poor designs and poor implementations (too much code for the functionality). This seems to be the case despite the best efforts of competent software engineers on the projects.

While I usually hesitate to ascribe to malice what may be due to incompetence, this effect is so widespread that I have to wonder. I note that only established large defense contractors can successfully bid on large defense contracts, and those contractors have many decades of experience with the government's understanding of software quality and its reaction to cost and schedule overruns. The contractors exist to make money, and the contracts are usually arranged so that the longer the contract takes, the more money the contractor makes. It may even be to the contractor's advantage to have the project fail (terminated by the customer before usable software is delivered). I have seen a contract in which the contractor received hundreds of millions of dollars before the contract was terminated. No failure ever seems to affect the contractors' ability to obtain new contracts. Such projects may be failures from the customer's point of view, but they are very successful from the contractor's.

I thus propose that the contractors have a highly optimized technique for finding the saddle point for maximizing profits while not angering the government enough to not win future contracts. Poor designs and implementations take longer to finish and to get working properly. Poor language choice would seem to be another tool to the same end.

Standard Languages: Java vs. Ada

From: aebrain@austarmetro.com.au
Date: Wed, 5 Sep 2001 09:38:59 +1000
Subject: Re: Standard Languages
To: team-ada@acm.org

[...] I agree with your main point, Java is not particularly standard, as anyone who's ever worked with it can attest.

Ada has 2 standards, Ada-83 and Ada-95.

Java has (so far), 1.0, 1.1, 1.2, 1.3 and the new still-not-stable 1.4. There is more difference between 1.0 and 1.1 than Ada-83 vs Ada-95. Then there's the Evil Empire(tm)'s Virtual Machine which is different again. Behaviour of any given Java source on various virtual machines is neither consistent nor easily predictable.

Now I like Java. After Ada, it's my favourite language, and I often find myself wishing "gee, I wish Ada could do X as quickly and easily as I can with Java 1.whatever". (Of course just as often, I wish that Java of any variety could do Y at all, when Ada-95 does it trivially, and Java suffers from fatal flaws due to its C ancestry, and .... but that's beside the point).

But Java standard? "The good thing about standards is that there are so many to choose from...".

From: Wes Groleau <wwgrol@ftw.rsc.raytheon.com>
Date: Wed, 5 Sep 2001 11:47:31 -0500
Organization: Raytheon Company
Subject: Re: Standard Languages
To: team-ada@acm.org

One thing I find amusing (in view of Java's alleged goal in life) is how often I hear of some hot new product in Java with "system requirements" listing a particular version of Windows. Once or twice, I thought "maybe that's a mistake from marketing and it really is portable." Each time, I found that it was not.

Wes Groleau,
http://freepages.rootsweb.com/~wgroleau

From: Michael Feldman <mfeldman@seas.gwu.edu>
Date: Wed, 5 Sep 2001 15:37:21 -0400
Subject: Re: Standard Languages
To: team-ada@acm.org

Remember "Java: write it once, run it anywhere."

Then later "Java: write it once, debug it everywhere."

Now I guess it's "Java: write it once, run it under Windows."

This industry finds it pretty hard to do anything meaningfully platform-independent.:-(

Java Often Misrepresented

From: James Rogers <jimmaureenrogers@worldnet.att.net>
Date: Mon, 08 Oct 2001 15:35:58 GMT
Subject: Re: is Ada dying?
Newsgroups: comp.lang.ada

> The Java classes are well documented. Much better than anything Ada has actually.

The Java classes are documented about as well as an Ada package specification documents an Ada package. I am speaking of the HTML API documentation generated from javadoc. Of course there are textbooks which expand on that documentation, just as there are Ada textbooks which expand on the information contained in the standard package specifications. [...]

> I find that I can much easier find a class in Java to do something, than I can find a function in Ada to do something. [...]

The javadoc tool is very useful. It localizes the documentation of the standard Java classes. The same can be said for the Ada RM. It is true that Java has more standard classes than Ada has standard packages. It is also true that all those standard Java classes are available to Ada compilers targeted at the JVM.

> Having a central single place to get things from is a Good Thing (tm). Examples of such things: want any Solaris package? http://www.sunfreeware.com/, want the JDK? http://java.sun.com, want the GNU Java collection? http://www.gnu.org/software/java/, want the Giant Java tree collection? http://www.gjt.org/, etc..

Wait a minute while I count my fingers and toes. It looks to me like this is a list of more than one place to find everything. Am I missing something here? [...]

Yes I am missing the concept that a list of four sites followed by "etc.." is exactly one. This is a feature of Java I have always found distasteful. The Java white papers, and subsequent Java supporters, have often made statements which are contrary to the normal usage of English. The example above declares that four or more sites is a single location. This is pure nonsense.

The Java white paper uses a lot of unsupported buzz words to describe Java. Some of my favorite are "simple" and "high performance". Java is not a simple language. There are thousands of standard classes to learn. Java is not high performance. It is simply faster than a dial-up network connection. What I am missing is an honest and accurate use of the English language.

> Show me an Ada site that is like java.sun.com. I know it is not fair for Ada to ask for this, given that even C++ does not have anything like that site, and C++ is much more used than Ada.

Well, I would say that adapower.com is pretty close. In fact, I believe adapower.com is a better starting point in a search for Ada information than java.sun.com is for Java information. There are relatively few links to non-Sun sites on java.sun.com.

> If you think the current state of Ada packages and libraries is good enough, I am happy for you. I am not arguing with you, [...]

You are now changing the subject. [...] The current state of Ada standard packages is very good. It is not as extensive as the set of Java standard classes. Quality and quantity are not the same thing.

For instance, Java provides several GUI packages useable in applets. The classes in the java.awt package hierarchy are useable in most browsers. The javax.swing classes are supposed to be useable in all browsers, but browser support for these classes varies. Note that applets (and servlets) also require you to learn another language, namely HTML. If you want to move to a more modern web server approach you can use JSP's, which require you to learn XML.

The biggest problem with browser support of Java applets is the differences in HTML required to support the Swing classes.

> Java has a standard, it is just not an ANSI nor ISO. But who cares. If you think having an ISO or ANSI stamp on the language will suddenly make it popular, then I am afraid you are completly wrong. Show me the VB standard out there, yet millions use VB to this day.

This is more Sun propaganda. Sun has a history of avoiding formal standards. They like to play in the arena of "defacto" standards. This means that they can produce a product and publish an API document for it. Once done, they call the product a standard.

Only Sun can decide what is Java and what is not. Only Microsoft can decide what is VB and what is not. This is the antithesis of open source. This is also forcing those using these tools to put complete trust in Sun and Microsoft. You have no input to the new features for the language. You only have the ability to report language defects if you pay for that privilege. [...]

[On "Given the changes in the language from Java 1.0 to Java 1.1 to JDK 1.2 to JSDK 1.3 to the almost released JSE 1.4, I wonder which language you use when you say you use Java." -- dc]

> [...] Java has improvements being added to it all the time. More packages and more libraries. You seem to think this is bad. [...] Generics are being now added to Java, and will be part of JDK 1.5. It is a good thing.

Generics may or may not be a good thing in Java. Interestingly, they will have a definite Ada flavor, rather than a C++ flavor. This is due in part to the fact that Norman Cohen has been actively involved in the definition and development of Java generics. [See also "Generics in Java" in AUJ 22.3 (September 2001), p.161. -- dc] I expect JDK 1.5 to be released some time in 2002. In terms of generics this will allow Java to catch up to Ada after a mere 20 years.

Note that up to this point Java supporters have been claiming that generics are unnecessary. They believed their inheritance model subsumed all requirements for generics. Could it be that they were wrong? There must be some reason for adding generics to JDK 1.5.

This is more evidence to me that Java is a language desparately working to live up to its press releases. For seven years Java has claimed the flexibility and extensibility provided by generics without having generics. Now they are adding generics to provide what is best provided by generics.

Similarly, Java has staunchly declared no need for a separation of specification and implementation. There have been several exceptions to this rule. You MUST create a Java interface to call a C library from Java. You MUST create a Java interface to create and deploy Enterprise Java Beans. You MUST create an interface to use the Java event model.

Such redefinitions of the language make programming in Java an adventure in learning.

Don't forget that you need standard patches to do some of the more useful stuff. For instance, you must patch the JSE 1.3 with JSEE 1.3 to be able to use Enterprise Java Bean technology. [...] You must first download J2SE 1.3, followed by J2SEE 1.3 if you want to use Enterprise Java Beans. You cannot simply download one or the other. [...] This means that your client's Java Runtime Environment must have the compatible libraries also. A big part of your Java system is shipped to your customers as the Java Runtime Environment. This presents you with serious compatibility and upgrade issues.

> I have no idea where you are coming on with all of the above. [...]

I have nothing against the use of Java in its appropriate domains. I do have issues with the way the Java community has misappropriated the English language. Many people use Java thinking they understand what is meant by that language. For instance, one local manager decided to re-write all his Cobol programs in Java. This meant retraining his entire IT staff. After making the decision he asked the question "What will be my performance improvement?"

Unfortunately the answer to that is about -30%.

Java and Real-Time

From: minyard@acm.org (Corey Minyard)
Date: Tue, 09 Oct 2001 03:42:59 GMT
Subject: Re: is Ada dying?
Newsgroups: comp.lang.ada

[In a thread where being suitable for "real-time" was confused with being "fast": -- dc]

Real-time does NOT mean fast. Real-time means guaranteed performance, like "I can stop the robot arm +/- 100us", or "the air bag will inflate between 1 and 1.5ms from impact". True real-time systems tend to have worse performance than non-real-time systems because providing the guarantees requires system overhead. As for performance, Java is fast approaching C/C++. Some tests we did on some platforms had C++ and Java within a few percent on just about everything.

Much theoretical work has been done on Java hard real-time performance, I have a copy of the spec, and it looks reasonable. You could build a moderately hard real-time system in Java, if you are willing to jump through all the right hoops. But the hoops are actually rather significant.

The main problem, though, is not with Java itself. If you use any third-party libraries, they will almost certainly violate your real-time constraints. So you can't use any third-party stuff in your system, or if you do, it has to be carefully isolated from the rest of your system. We used several third-party libraries in our system. They all were extremely sloppy with memory management; they threw tons of garbage needlessly. Plus, because of the slack Java package/class usage rules, Java software tends to be a "big glob of software", you generally cannot extract just the parts of the code you need because everything uses everything else. Because of lazy initialization rules, it would be easy for hitting a new path in the code to cause a mass initialization event.

So the bottom line, IMHO, is that it's not worth it to do hard real-time in Java. The big advantage of Java is all the stuff that comes along with it, but you really can't use that stuff in a real-time application. To get true real-time in Java, you have to manage your own memory and segment your application. You have to be very careful with garbage generation. But if that's the case, why not write the real-time portion in another language and interface it with Java?

Note that soft real-time is a different story. You can probably implement a soft real-time system in Java if you are willing to do some work. Before what I was working on was cancelled, we had a reasonable system working, and I knew of others that had at least limited success. But I know of no practical written material on this subject, and I doubt any exists, because it's kind of a black art right now. And we did a lot of customization to our chosen compiler to help us meet our goals, including a custom GC and some careful analysis on how the libraries worked. So it's still not easy, but it might be better than using C/C++ due to Java's improved safety. I have a lot of knowledge on the subject, but in essence it's only theoretical because we never actually delivered a product.

But then, I'd rather use Ada. It has all the safety advantages of Java (and more) without the baggage of GC, lack of call-by-reference, etc. But non-technical reasons often take precedence. We did a language analysis on the project in question. Ada won by a significant margin, but we chose Java anyway.

The Origin of Java

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Wed, 10 Oct 2001 12:04:10 -0400
Subject: Re: Gratuitous bashing ?
Newsgroups: comp.lang.ada

> http://www.cafeaulait.org/1998august.html Look for the word "Ada" within it....

I noticed they claim Java was developed for embedded systems. I don't recall that being the objective behind Java [...]. Last I heard, it basically went to interpreted byte code - which is not a typical strategy for embedded computing. This is rather implying that the author might not be up on all the facts about Java and hence might not be the best source for information about Ada (which *was* developed for embedded systems.)

From: Darren New <dnew@san.rr.com>
Date: Wed, 10 Oct 2001 16:37:12 GMT
Subject: Re: Gratuitous bashing ?
Newsgroups: comp.lang.ada

IIRC, Java was originally called "Green" or "Greentree" or some such, and was intended for set-top boxes. The idea was to allow stuff like TV listings and such to not rely on the specifics of the set-top box. [...]

From: Wes Groleau <wwgrol@sparc01.ftw.rsc.raytheon.com>
Date: Wed, 10 Oct 2001 15:52:48 -0500
Organization: Raytheon Company
Subject: Re: Gratuitous bashing ?
Newsgroups: comp.lang.ada

> I don't know that Java was invented *specifically* for STBs - I rather recall it being something developed because they thought web apps would be multi-platform....

According to a Sun marketing video, a group was assigned to develop a set top box or something. This part of the video was very unclear. I had the impression that it was something like WebTV.

Anyway, they were working in C, and kept having problems traced to certain kinds of mistakes. So they invented a language in which you could not make those kinds of mistakes.

Joy or Gosling (I forgot which, they were both in the video) held up a book on C with lots of stuff lined out and said something like, "Basically, we just went through here and deleted everything that was causing us problems."

Then somebody else saw it and thought, "Hey, we could sell this!"

Really. I saw it. It really was like that. Honest.

OK, my bias may have colored my report a little--but not much!

From: Wes Groleau <wwgrol@sparc01.ftw.rsc.raytheon.com>
Date: Thu, 11 Oct 2001 10:20:46 -0500
Organization: Raytheon Company
Subject: Re: Gratuitous bashing ?
Newsgroups: comp.lang.ada

> O.K. I'll accept that story. However, that sounds more like Necessity being the Mother of Invention. [...], it doesn't sound like somebody said: "Since we have to program STBs, let's design a language that is suited to STB software development..." That seems different than "Lets use a subset of C because we are having trouble with full-up C... Hey wait a minute! This is a new language!!!!"

I left out the part where they did add some OO features (with as little true engineering as the rest of it).

I also left out my own observation that they did _not_ remove some bad features of C because, being more experienced, they did not have as many problems with those features. They also removed _good_ features "because they are unsafe" which Ada and other languages had already proven can be done safely.

Java and Teaching Programming

From: john.mccabe@emrad.com (John McCabe)
Date: Mon, 15 Oct 2001 08:21:42 GMT
Organization: Emrad Ltd
Subject: Re: is Ada dying?
Newsgroups: comp.lang.ada

> (In fact I even think that using Pascal is better than using Java as a first language).

Of course it is - Pascal was designed as a teaching language, Java wasn't designed.

From: john.mccabe@emrad.com (John McCabe)
Date: Mon, 15 Oct 2001 16:10:52 GMT
Organization: Emrad Ltd

Subject: Re: is Ada dying?
Newsgroups: comp.lang.ada

> did you mean to cut off the sentence above as is,

Yes.

> i.e. did you mean to say that Java was not designed, period, or that it was not designed to be a teaching language?

The former.

> [...] so many do not learn the more basic things in programming, like data structures and records (in Java, programmers do not even know what a record is :). These things are not learned well. Java programmers do not even know too well about enumeration and parameters passing mehanism, but know how to create an object or extend one and use an interface.

Enumeration types are an almost mandatory feature of any language that can claim to be safe but, even then, it depends on the implementation. Ada's definition of Enumeration types, and their use as array and loop bounds is excellent - the C++ version is vaguely useful, but not all that good relying too much on history. The fact that Java does not even have Enumeration types is, to me, a serious defect.

> Speaking of records. In true OO, the concept of a record does not exist really. All what you have is an object, which contains attributes (state information). So, I can sort of understand when a Java programmer ask me what is a record?

I can understand that and, often, a record can be replaced by a Class with no behavioural aspects (i.e. no methods). The problem is that the overhead (possibly just in syntax) of a class can be a nuisance and confuse the issue and, certainly in Java, there is no way to define the representation of elements in a Class (unlike Ada with its very powerful representation clauses). To me this shows a great difference in the target audience of the languages, and their history.

Security Issues

From: Tom Moran <tmoran@acm.org>
Date: Thu, 4 Oct 2001 19:43:38 GMT
Subject: 9/11 -> Ada
To: team-ada@acm.org

Seems to me a responsible person trying to make important software less vulnerable to "cyberterrorism" must seriously consider Ada as one of his tools. Using known-to-be-fragile tools would be irresponsible.

From: Carlisle Martin C Dr USAFA/DFCS <Martin.Carlisle@usafa.af.mil>
Date: Fri, 5 Oct 2001 06:13:50 -0600
Subject: Re: 9/11 -> Ada
To: team-ada@acm.org

> Interesting theory, but nobody is really exploiting language flaws to break into security systems. If Ada has built in security measures that C++ doesn't (and Java does) then that is useful to sell, but security is much more than preventing overflow on an array.

I agree and disagree. The vast majority of security flaws I see are overflowing arrays. These would all go away in Ada or Java. However, certainly there are other flaws for which language is not a magic bullet.

Martin C. Carlisle, PhD, Associate Professor of Computer Science, United States Air Force Academy

From: Wesley_Groleau@raytheon.com
Date: Fri, 5 Oct 2001 09:10:54 -0500
Subject: Re: 9/11 -> Ada
To: team-ada@acm.org

> [...] nobody is really exploiting language flaws to break into security systems. [...]

Seems to me that many of the malicious hacks and viruses (but certainly far from all) DID depend on flaws in C, especially lack of bounds checking. I have heard plenty about flaws in Java and/or JVM security, but I have never heard of an actual break-in using such a flaw.

The original post hinted at cyber-terrorism. Perpetrators of such an act are concerned with causing damage, not with getting anything out of it. Therefore, they do not have to be concerned with "doing things neatly."

From: Michael Feldman <mfeldman@seas.gwu.edu>
Date: Fri, 5 Oct 2001 12:24:28 -0400
Subject: Re: 9/11 -> Ada
To: team-ada@acm.org

> [...] many of the malicious hacks and viruses [...] DID depend on flaws in C, especially lack of bounds checking.

And an interesting paper a while back in CACM reported on a world-wide survey of people's nastiest bugs, and more than 50% were memory-related, things like array overrunning, trash pointers, etc.

BUT... any language or environment that allows arbitrary access to memory or files can be used to do damage. Let's not oversell Ada on "security" grounds - anyone smart enough to use pragma Suppress can get around all our wonderful language features. I content that all our checking was really designed to help us prevent _bugs_, not _mischief_. One could envision a specific compiler and/or execution environment, designed for safety, that prevented such arbitrary access, and ignored (or rejected) Suppress, but that's a tool issue, not a language one.

> [...] flaws in Java and/or JVM security, [...]

Well, as long as there are flaws, someone will be smart and mischievous enough to exploit them. Flaws and bugs can (should) be fixed.

[...] Some of the recent worms ("I Love You", etc.) exploited "cool features" of Microsoft products, like the ability to run VB scripts from an e-mail attachment. I've read some fairly scary trade-press articles about M$'s seemingly contemptuous attitude toward security. I'm actually quite surprised that we haven't seen any multi-billion-dollar class-action suits about this. Where are the lawyers when we need them? :-)

[Wesley_Groleau@raytheon.com responded: -- dc]

Nor do they learn from history--nearly twenty years ago VMS began filtering escape sequences out of e-mails because besides allowing fancy formatting (can you say HTML?) they were being used to reprogram the function keys on sysadmin's terminals to execute commands.

[And after yet another mail arrived with a worm attached, Nick Roberts wrote: -- dc]

Good Lord, isn't it just Twighlight Zone that IIS still suffers from 'buffer overrun' weaknesses (yes that's weaknesses plural)? Write it in Ada and they vanish! How's that for Ada advocacy?

From: Wesley_Groleau@raytheon.com
Date: Fri, 5 Oct 2001 11:46:07 -0500
Subject: Re: 9/11 -> Ada
To: team-ada@acm.org

> [...] Let's not oversell Ada on "security" grounds - anyone smart enough to use pragma Suppress can get around all our wonderful language features. [...]

I agree with the selling part and the "bugs not mischief", but if you're a cracker, you can't add pragma suppress to your target's code and recompile/reinstall it. So you instead attack programs written in a language that doesn't allow suppress to be turned off. :-) [...]

From: Tom Moran <tmoran@acm.org>
Date: Sun, 7 Oct 2001 13:23:16 -0700
Organization: Decision Aids
Subject: Re: 9/11 -> Ada
To: team-ada@acm.org

We agree security is not just preventing array overflows, and Ada is not a magic bullet. But the environment has changed. The estimated cost of a bug in infrastructure software has increased because we now realize it has a higher likelihood of being exploited by bad guys and the cost of damage therefrom is higher than we thought. So "C's fine, why use Ada?" should change to "How can you justify not using better tools (Ada among them)".

Another change is the drop in interest/discount rates. That means long term savings are now raised in importance relative to short run costs. "How can you justify ignoring the long term, life cycle, savings of using Ada?"

As to whether Ada is indeed better for bug prevention and long term maintenance, "prove it" gives way to "what evidence exists says Ada's better - if you don't think so, prove that."

How to Stamp Out Buggy Software?

From: "Thomas A. Panfil" <t.panfil@gte.net>
Date: Wed, 10 Oct 2001 22:23:10 -0400
Subject: Request for Comments on How to Stamp Out Buggy Software
To: team-ada@acm.org

Journalist seeks comments on how to stamp out buggy software!!

See article in 01 Oct 2001 issue of "Network Computing" entitled: "Growing Up with a Little Help from the Worm". It's in a "Security Watch" column. See URL: http://www.networkcomputing.com/1220/1220colshipley.html

It ends with the following (note the nice way he starts his comment): "... The time is right for our industry to get educated and smart -- and that means we all have to grow up. Send your comments on this column to Greg Shipley at gshipley@neohapsis.com."

Perhaps some Team-Ada members can provide useful comments to him. [...]

Doesn't Management Like Ada?

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Tue, 23 Oct 2001 09:48:05 -0400
Subject: Re: Have you ever had a bug caused by...
Newsgroups: comp.lang.ada

> But if instead of chasing bugs you could add new features which would improve the bussiness of the companies that buy the product... :-)

What always confounds me about it is that even when you can make a strong business case for using Ada based on higher reliability, lower development and maintenance cost, etc., and actually back it up with data, you still lose because management will end up asking their techies about it (fair enough) and the techies don't like Ada or don't want the headache of switching to Ada, so they recommend against it. A manager can't be expected to know all the technical aspects and they've got to trust their techies to make those decisions, so I don't really blame them. That's why it is important to make the case with the techies and get them to want to use Ada. [...]

From: Ted Dennison <dennison@telepath.com>
Date: Tue, 23 Oct 2001 15:45:24 GMT
Subject: Re: Have you ever had a bug caused by...
Newsgroups: comp.lang.ada

Actually, I've seen more cases where the grunts like it, but management doesn't for whatever reason (I quit trying to analyze their reasoning process when I discovered it has little to do with reasoning).

From: "Marin David Condic" <marin.condic@pacemicro.com>
Date: Tue, 23 Oct 2001 13:08:07 -0400
Subject: Re: Have you ever had a bug caused by...
Newsgroups: comp.lang.ada

This is sometimes a problem as well. All too often some (pointy haired) bosses have been busy reading industry journals and pick up on buzzwords and think "this is the direction to go in" because everybody is writing about it. Trade journals can suffer from the same GIGO phenomenon as computers: Garbage In, Gospel Out.

We used to have an "insider" expression taken from a Dilbert commic: "Mauve has more RAM..." to indicate when a manager was speaking techno-babel. It had to do with Dilbert testing to see if the PHB knew what he was talking about when he wanted to have Dilbert acquire a new relational database. Dilbert asks what color it should be. The answer was mauve because "Mauve has more RAM".

The sad thing about Dilbert is that Scott Adams is *not* making this stuff up.

From: "Marc A. Criley" <mcqada@earthlink.net>
Date: Wed, 24 Oct 2001 12:44:14 GMT
Organization: Quadrus Corporation
Subject: Re: Have you ever had a bug caused by...
Newsgroups: comp.lang.ada

When mentioning Ada to some developers where I used to work that were working on C++ projects, I found that a number of them had used Ada in the past and liked it. But then, when new projects started up with C++ as the implementation language, they, being good engineers, simply learned the new language and went to work.

The _strident_ calls for C++ (and now Java) often seemed to come from a small number of opinionated individuals who were only too happy to get up in front of management or the customer and beat the pulpit on how "we have to go where the market is going!". The majority of these individuals were on-staff types who hadn't coded in years... but certainly knew how it ought to be done!

I had a a perpetual smouldering battle against one such faction on the project on which I worked, who advocated tossing out a measurably high-quality, efficient, reliable, maintainable weapon control simply because it was written in Ada, and redo it in C++.

In a follow-on program, I knew that there was no chance of any new subsystems being written in Ada, but these techno-surfers were advocating writing a soft realtime mission- and safety-critical portion of the system in Java--and not today's Java, but 1999 Java. Even the skeletal version done for the proposal demo was unstable and a total dog on performance. I went ballistic over that and the blind advocacy fell appart, with the system ending up being proposed in C++. (And I knew the lead designer was a very competent engineer, so I had less heartburn than I might've otherwise.)

Marc A. Criley, Senior Staff Engineer, Quadrus Corporation, www.quadruscorp.com

From: Jeffrey Carter <jeffrey.carter@boeing.com>
Date: Wed, 24 Oct 2001 18:55:30 GMT
Organization: The Boeing Company
Subject: Re: Have you ever had a bug caused by...
Newsgroups: comp.lang.ada

> [...], when new projects started up with C++ as the implementation language, they, being good engineers, simply learned the new language and went to work.

When new projects started with substandard steel as the implementation material, they, being good civil engineers, simply built the bridge with it.

Of course, when the bridge collapses and kills people, the civil engineers go to jail. When the S/W fails, the manager gets promoted.

Tony Hoare on Ada - a Quote

From: mjsilva697@earthlink.net (Mike Silva)
Date: 14 Oct 2001 10:27:11 -0700
Subject: Re: Language design by by committee
Newsgroups: comp.lang.java.advocacy, comp.lang.ada

[In a thread where once more the old urban legend circulated about "Tony Hoare was an early member of the [Ada] design committee until he left in disgust. In his ACM Turing Award lecture "The Emperor's Old Clothes" he says just that." -- dc]

> Tony Hoare left because he thought the language was way too big. Later on he wrote a foreward to a book in which he recanted this view, and hope people would have the opportunity to use Ada [...]

A Google search finds this:

Here is the text of Mr. Hoare's foreword to an Ada book in 1987. You may judge for yourself whether he was as anti-Ada as you suggest.

C.A.R. Hoare's comments in the foreward to Ada Language and Methodology

" 'I enjoyed reading the Algol 60 report; it taught me a lot about programming.' This is the comment of a data processing manager of a major motor manufacturing company, who had no conceivable prospect of ever using the language to program a computer. It is a most perceptive comment, because it describes an important goal in the design of a new programming language: that it should be an aid in specification, description, and design of programs, as well as in the construction of reliable code.

This was one of the main aims in the design of the language which was later given the name Ada. As a result, the language incorporates many excellent structural features which have proved their value in many precursor languages such as Pascal and Pascal Plus.

The combination of many complex features into a single language has led to an unfortunate delay in availability of production-quality implementations. But the long wait is coming to an end, and one can now look forward to a rapid and widespread improvement in programming practice, both from those who use the language and from those who study its concepts and structures.

I hope that this book will contribute directly to these ideals, which have inspired many of the other books in the same series. It continues the tradition of the series in that it describes how the language can be used as the target of sound programming methodology, embracing the full life-cycle of a programming project. It explains not just the features and details of the language, but also their purpose and method of effective use.

The complexities and difficulties are not glossed over; they are explained within the appropriate context, with hints on how to avoid any consequent problems. I hope the book will be useful, both to those who have the privilege or obligation to use the language, and to those who have the interest and curiosity to understand and appreciate its rationale."

from the foreword to "Ada Language and Methodology", David A. Watt, Brian A. Wichmann, and William Findlay, Prentice-Hall International Series in Computer Science, ISBN 0-13-004078-9, Published in 1987.

[Next time you hear or read this urban legend, I suggest you use the above extract. -- dc]

Conference Calendar

This is a list of European and large world-wide events that may be of interest to the Ada community. More information on items marked ♦ is available elsewhere in the *Journal*. The information here is extracted from the online *Conference announcements for the international Ada community* at http://www.cs.kuleuven.ac.be/~dirk/ada-belgium/events/list.html on the Ada-Belgium webserver. These pages contain full announcements, calls for papers, calls for participation, programmes, URLs etc and are updated regularly.

2002

07-09 January	**7th IEEE Computer Society's International Workshop on Object-oriented Real-Time Dependable Systems (WORDS'2002)** San Diego, California, USA.
07-10 January	**2002 Embedded & Real-time Distributed Object Systems Workshop (RTEmbedded'2002)** Burlingame, California, USA.
16-18 January	**29th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'2002)** Portland, Oregon, USA.
19 January	**9th International Workshop on Foundations of Object-Oriented Languages (FOOL 9)** Portland, Oregon, USA. Topics include: language semantics, type systems, program analysis and verification, concurrent and distributed languages, etc.
21-24 January	**Software Engineering with SPARK Training Course** Bath, UK. Topics include: in addition to the normal content, these courses will cover the new and improved features of Release 6.0 of the SPARK Toolset.
22 January	**Workshop on Refinement of Critical Systems: Methods, Tools and Experience (RCS'2002)** Grenoble, France. Topics include: Refinement applied to software engineering, control systems, distributed systems, embedded systems, real-time, reactive and hybrid systems, information systems; etc.
23-25 January	**8th International Conference on Languages and Models with Objects (LMO'2002)** Montpellier, France. Topics include (in French): Programmation par objets (Languages, interpretation, compilation; modeles d'objets pour la programmation; objets et types; environnements de programmation; etc.); Composants et objets en reseau (Modeles de composants a objets; interactions de composants; developpement a base de composants, composants reutilisables; objets et composants distribues, repartis; acteurs, parallelisme; objets et internet; interoperabilite); Genie des objets (Cycle de vie des objets; retro-conception, evolution des programmes, versions; surete des programmes, specifications formelles; methodes d'analyse et de conception objet, UML; ingenierie des modeles et des meta-modeles; reutilisation, architectures logicielles reutilisables et a base de composants; hierarchies, frameworks, patterns); Applications; etc.
18-21 February	**14th Software Engineering Process Group Conference (SEPG'2002)** Phoenix, Arizona, USA.
18-21 February	**Technology of Object-Oriented Languages and Systems (TOOLS Pacific'2002)** Sydney, Australia. Theme: "Objects for Internet, Mobile, and Embedded Applications".
21-22 February	**2nd Workshop on Aspect-Oriented Software Development (AOSD'2002)** Bonn, Germany.
25-26 February	**Workshop on Open Source Software Development** Newcastle upon Tyne, UK.
25-27 February	**15th Conference on Software Engineering Education and Training (CSEET'2002)** Covington, Kentucky (Greater Cincinnati), USA.
TBD March	**1st Bi-Annual Embedded Systems Club Conference** Newbury, UK. Topics include: knowledge of technologies and practices related to embedded systems engineering; user experience with embedded technologies or practices applied to industrial strength applications; emerging embedded technologies; etc.

04-08 March **International Conference on Practical Software Quality Techniques & Testing Techniques (PSQT/PSTT'2002 South)** New Orleans, USA. Deadline for early registration January 25.

06-08 March **Ada-Deutschland Tagung 2002** Jena, Germany. Topics include (in German): Methoden und Werkzeuge für Echtzeitsysteme; Qualitätsmanagement in SW-Projekten; Vorgehensmodelle und Lifecycle Management von IT-Systemen mit Ada; Echtzeitsysteme mit Ada (Annex D); Interoperabilität von Ada und anderen Programmiersprachen; Sichere Software mit Ada (Annex H); Erfahrungsberichte über Produktivität, Performance und Kosten in Ada-Projekten; Ada in der Ausbildung; etc..

10-13 March **2002 ACM Symposium on Applied Computing (SAC'02)** Madrid, Spain.

11-13 March **6th European Conference on Software Maintenance and Reengineering (CSMR'2002)** Budapest, Hungary.

11-13 March **5th International Conference on "Achieving Quality In Software" (AQUIS'2002)** Venezia, Italy.

11-15 March **5th International Internet & Software Quality Week Europe (QWE'2002)** Brussels, Belgium QWE'2001 was rescheduled from 12-16 November 2001 to 11-15 March 2002. Theme: "Internet NOW!" Description: QWE2002 focuses on advances in software test technology, reliability assessment, software quality processes, quality control, risk management, software safety and reliability, and test automation as it applies to client-server applications and to websites. Topics include: Productivity and Quality Issues; Process Improvement; Real-Time Software; Object Oriented Testing; Application of Formal Methods; Cost/Schedule Estimation; Software Reliability Studies; E-Commerce Reliability; Quality of Service (QoS); Risk Management; etc. Deadline for early registration: January 25, 2002.

14-15 March **2nd International Software Process Improvement and Capability determination Conference (SPICE'2002)** Venice, Italy.

20-22 March **5th IFIP International Conference on Formal Methods for Open Object-based Distributed Systems (FMOODS'2002)** Twente, The Netherlands. Topics include: Specification and analysis techniques for distributed systems; Semantics of object-based programming languages; Design and software life-cycle of object-based distributed applications; Applications to telecommunications and related areas; etc.

06-14 April **European Joint Conferences on Theory and Practice of Software (ETAPS'2002)** Grenoble, France. Includes:

- 06-14 April **8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2002)** Topics include: Verification and construction techniques; Compositional and refinement-based methodologies; Analytical techniques for real-time, hybrid and safety-critical systems; Tool environments and tool architectures; Applications and case studies; etc..
- 07 April **Workshop on Software Composition (SC'2002)**
- 07 April **International Conference on Compiler Construction (CC'2002)** Topics: compiler construction, programming language implementation and language design.
- 08-12 April **Fundamental Approaches to Software Engineering (FASE'2002)** Topics include: Experience reports on best practices with component models and specifications, development tools, modelling environments, and software development kits; Integration of formal concepts and current best practices concepts in industrial software development; etc.
- 13 April **2nd Workshop on Language Descriptions, Tools and Applications (LDTA'2002)**
- 13 April **Synchronous Languages, Applications, and Programming (SLAP'2002)** .

07-10 April **10th Object Technology Conference (OT'2002)** Oxford UK. Topics include: Component technology, Languages, Distributed systems, Small and embedded systems, Patterns, Lessons learned/experience reports etc.

08-10 April **6th International Conference on Empirical Assessment and Evaluation in Software Engineering (EASE'2002)** Keele University, UK. Deadline for submissions: February 7, 2002 (experience reports), April 2, 2002 (posters).

08-11 April **Software Engineering with SPARK Training Course** Bath, UK Topics include: in addition to the normal content, these courses willcover the new and improved features of Release 6.0 of the SPARK Toolset.

08-11 April **9th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS'2002)** Lund, Sweden. Topics include: Component-based Design and Reuse; Applied Formal Methods and Security; Tools and Environments; Education and Training; Embedded Systems; Reliability, Dependability, Safety; Verification and Validation; Standards; etc.

09-12 April **11th International Real-Time Ada Workshop (IRTAW'2002)** Mont-Tremblant, Quebec, Canada.

15-17 April **9th Annual European Concurrent Engineering Conference (ECEC'2002)** Modena, Italy Topics include: Formal Methods and Techniques; Engineering of embedded systems (e.g. HW/SW co-design, system development process, specification languages, ...); Networking and distribution in CE (e.g. CORBA based environments and integrated frameworks, Architectures for building CE systems, ...); Practical Applications and Experiences (e.g. Practical solutions, Pitfalls and success stories, Case studies, pilot projects and experiments, ...); etc.

15-19 April **7th International Conference on Software Reuse (ICSR-7**) Austin, Texas, USA. Topics include: Software product lines and product line architectures; Component-based software engineering; Lightweight approaches to software reuse; Quality aspects of reuse, e.g. security and reliability; Success and failure stories of reuse approaches from industrial context; etc. Deadline for submissions: January 30, 2002 (posters and demos).

15-19 April **International Parallel and Distributed Processing Symposium (IPDPS'2002)** Fort Lauderdale, Florida, USA. Topics include: Applications of parallel and distributed computing, including web applications and scientific applications; Parallel and distributed software, including parallel programming languages and compilers, operating systems, schedulers, runtime, middleware, libraries, programming environments and tools for parallel and distributed computing; etc. Includes:

15-19 April **7th International Workshop on Formal Methods for Parallel Programming: Theory and Applications (FMPPTA'2002)**

17-19 April **3rd International Conference on Software Testing (ICSTEST'2002)** Dusseldorf, Germany.

21-24 April **2nd International Conference on Computational Science (ICCS'2002)** Amsterdam, The Netherlands. Topics include: Parallel and Distributed Computing; Problem Solving Environments (including: Software Component Technology); Education in Computational Science; etc.

23-26 April **1st International Conference on Aspect-Oriented Software Development (AOSD'2002)** Enschede, The Netherlands. Topics include: language design and implementation; analysis, design and development tools; software engineering; lifecycle support; etc. Deadline for submissions: January 15, 2002 (demonstrations).

29 April – 01 May **5th IEEE International Symposium on Object-oriented Real-time distributed Computing (ISORC'2002)** Washington DC, USA.

15-18 May **3rd International Conference on Integrated Formal Methods 2002 (IFM'2002)** Turku, Finland.

19-25 May **International Conference on Software Engineering (ICSE'2002)** Buenos Aires, Argentina.

26-29 May **3rd International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP'2002)** Alghero, Sardinia, Italy.

27-31 May **14th Conference on Advanced Information Systems Engineering (CAiSE'02)** Toronto, Canada. Topics include: Distributed, Web and Mobile Architectures; OO and Agent-Oriented Technologies and their Applications to IS Development; Languages and Protocols for IS; Component-ware and IS; etc. Deadline for submissions: March 1, 2002 (posters).

04-07 June **8th International Symposium on Software Metrics (Metrics'2002)** Ottawa, Canada Theme: "Measuring and Managing Software Risks in the Age of Internet".

09-12 June **7th European Conference on Software Quality** Helsinki, Finland.

09-14 June **27th Annual USENIX Technical Conference (USENIX'2002)** Monterey, Canada. Topics include: Reliability and QoS; Usage studies; Web technologies; Interoperability of heterogeneous systems; special track on freely redistributable technology (GNOME, GNU, Linux, Tcl/Tk and more); etc.

10-14 June **16th European Conference on Object-Oriented Programming (ECOOP'2002)** Málaga, Spain. Topics include: implementations of language features; language support for security and safety; techniques for embedded and mobile code; compilation for distributed, heterogeneous systems; languages and compilers for parallel computing; etc. Deadline for submissions: April 15, 2002 (demonstrations, posters)

17-19 June **ACM SIGPLAN 2002 Conference on Programming Language Design and Implementation (PLDI'2002)** Berlin, Germany. Sponsored by ACM SIGPLan in cooperation with ACM SIGSoft Topics include: implementations of language features; language support for security and safety; techniques for embedded and mobile code; compilation for distributed, heterogeneous systems; languages and compilers for parallel computing; etc.

♦ 17-21 June **7th International Conference on Reliable Software Technologies - Ada-Europe'2002** Vienna, Austria. Sponsored by Ada-Europe, in cooperation with ACM SIGAda. Topics include: management of software development and maintenance; software quality; software development methods and techniques; software architectures; tools; kinds of systems; applications; Ada language and tools; Ada experience reports; education and training; case studeies and experiments; and a special session on embedded systems, including the use of Ada in this realm.

19-21 June **ACM SIGPLAN Joint Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'02) and Software and Compilers for Embedded Systems (SCOPES'02)** Berlin, Germany. Immediately after PLDI'02. Topics include: Programming languages for embedded applications; Software design for multiprocessor systems; Memory management/garbage collection for embedded systems; Concurrent+distributed embedded environments/runtime systems; Real-time operating systems: environment and tools (e.g., RT-Linux); Exception and interrupt handling for real-time; Code generation for embedded processors; Program optimization for real-time performance and DSPs; Real-time scheduling analysis; etc. Deadline for paper submissions: February 1, 2002.

20-21 June **1st International IFIP/ACM Working Conference on Component Deployment (CD'2002)** Berlin, Germany.

23-26 June **International Conference on Dependable Systems and Networks (DSN'2002)** Washington, D.C., USA. Deadline for submissions: January 14, 2002 (tutorials).

24-27 June **2002 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'2002)** Las Vegas, Nevada, USA. Topics include: Parallel/Distributed applications; Reliability and fault-tolerance: Software and hardware fault-tolerance (system- and application-level), etc.; Real-time and embedded systems; Object Oriented Technology and related issues; Software tools and environments for parallel and distributed platforms: Operating systems, compilers, languages, debuggers, monitoring tools, software engineering on parallel/distributed systems, ...; Education: parallel and distributed processing in computer science curriculum (both graduate and undergraduate levels.); Recent history (last decade) of parallel/distributed processing and what to expect during the next decade if history repeats itself; etc. Deadline for submissions: February 22, 2002 (draft papers).

02-03 July **International Workshop on Distributed Event-Based Systems (DEBS'02)** Vienna, Austria. Topics include: Programming language support and integration (e.g. typing, abstractions); Real-

time distributed event systems; Integration with standard middleware; Fault-tolerant event distribution; Quality of service and its specification; Case studies of challenging applications and requirement analysis; etc.

20-24 July **11th Formal Methods Europe Symposium (FME'2002)** Copenhagen, Denmark. Theme: "Formal Methods: Getting IT Right". In conjunction with the third Federated Logic Conference (FLoC'02) Deadline for submissions: January 15, 2002.

20-23 August **13th International Conference on Concurrency Theory (CONCUR'2002)** Brno, Czech. Repubic. Topics include: concurrency related aspects of: real-time systems, distributed programming, object-oriented programming, case studies, tools and environments for programming and verification, etc. Deadline for submissions: December 1, 2001 (workshops), March 25, 2002 (papers)

26-28 August **International Conference on Pervasive Computing (PERVASIVE'2002)** Zurich, Switzerland. Deadline for submissions: February 22, 2002 (papers and demos), June 19, 2002 (posters and short papers).

27-30 August **European conference on Parallel Processing (Euro-Par'2002)** Paderborn, Germany. Topics include: Support Tools and Environments; Performance Evaluation, Analysis and Optimization; Distributed Systems and Algorithms; Parallel Programming: Models, Methods and Programming Languages; etc. Deadline for submissions: February 8, 2002.

09-12 September **7th International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems (FTRTFT'2002)** University of Oldenburg, Germany. Deadline for submissions: March 1, 2002.

10-13 September **21st International Conference on Computer Safety, Reliability and Security (Safecomp'2002)** Catania, Italy. Focuses on safety-critical computer applications. Deadline for submissions: February 10, 2002 (papers), April 7, 2002 (tutorials).

17-20 September **6th International Enterprise Distributed Object Computing Conference (EDOC'2002)** Lausanne, Switzerland. Deadline for submissions: March 4, 2002 (abstracts), March 22, 2002 (papers), April 15, 2002 (tutorials).

23-27 September **17th IEEE International Conference on Automated Software Engineering (ASE'2002)** Edinburgh, U.K. Deadline for submissions: May 6, 2002 (abstracts), May 13, 2002 (papers).

TBD September **2002 ACM SIGAda Annual International Conference (SIGAda'2002)** Houston, Texas, USA.

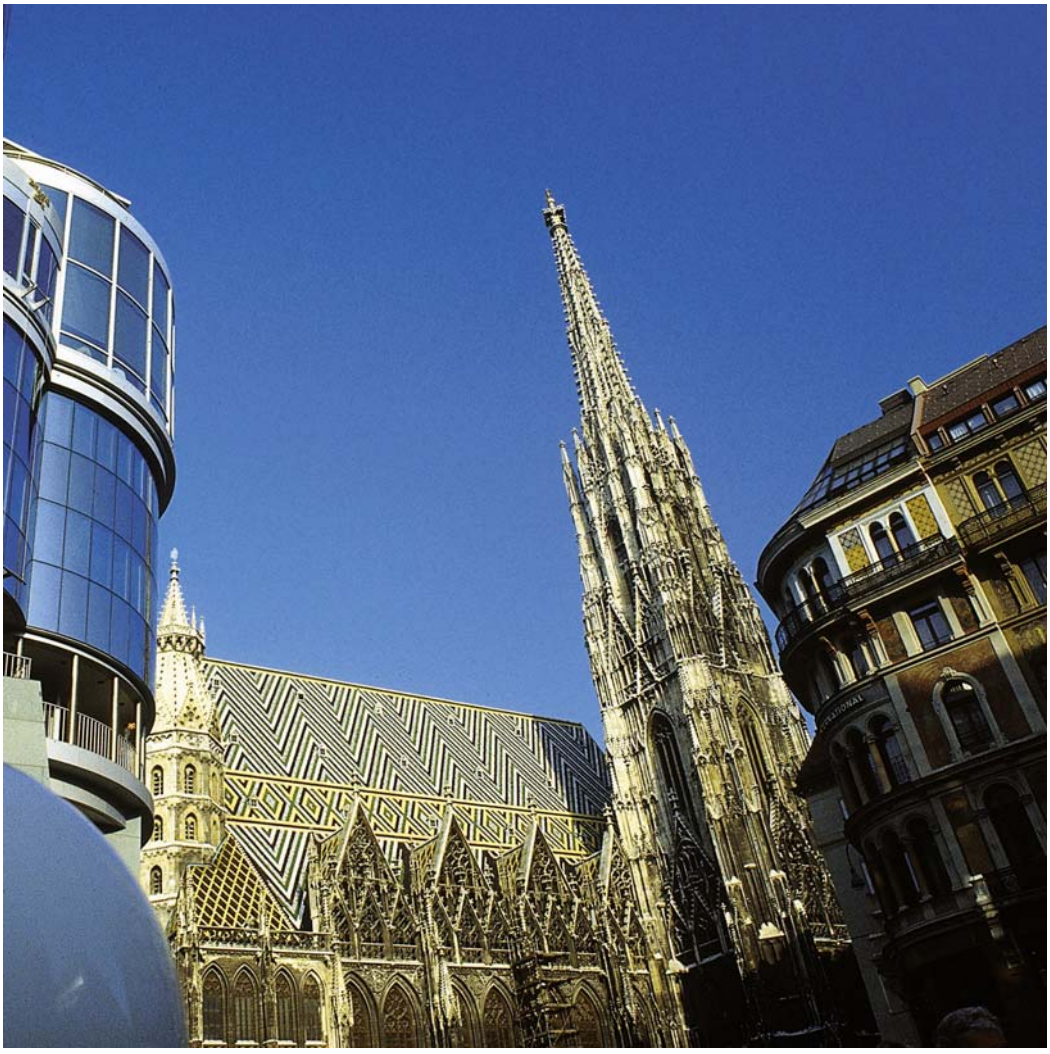
09-11 December **5th USENIX Symposium on Operating Systems Design and Implementation (OSDI'2002)** Boston, Massachusetts, USA. Topics include: distributed systems, parallel systems, embedded systems, the influence of hardware development on systems and vice-versa, etc. Deadline for paper submissions: May 12, 2002.

10 December **Birthday of Lady Ada Lovelace, born in 1815 – Happy Programmers' Day!**

2003

05-13 April **European Joint Conferences on Theory and Practice of Software (ETAPS'2003)** Warsaw, Poland. Event includes: conferences from 7 to 11 April 2003, affiliated workshops on 5-6 and 12-13 April, 2003.

7th International Conference on Reliable Software Technologies - Ada-Europe 2002



Vienna, Austria, June 17-21, 2002

http://www.ada-europe.org/conference2002.html

SIGAda

TECHNISCHE UNIVERSITÄT WIEN

Ada-Europe 2002

In 2002, the 7th International Conference on Reliable Software Technologies will take place in Vienna, Austria, from June 17th to June 21st. The conference offers a technical program and exhibition, plus a series of tutorials and a workshop.

The conference provides an international forum for researchers, developers and users of reliable software technologies. Presentations and discussions cover applied and theoretical work currently conducted to support the development and maintenance of software systems.

Vienna, a city with about 2 million inhabitants is situated in the heart of Europe. It is a city on which its ever-changing history has left an indelible mark, manifested also in the rich cultural heritage. Shaped by its hundreds of years as capital of an empire, the city's ultimate fascination nowadays stems from combining imperial grandeur with explosive modernity.

The conference will take place in the Parkhotel Schönbrunn which originated in 1907 as the guest house of Emperor Franz Josef I. The newly renovated hotel is located in the immediate vicinity of the "Schönbrunn Palace" and its beautiful surrounding park, situated close to the center of Vienna.

Preliminary Program

<table>
<tr><th></th><th>Morning</th><th>Late Morning</th><th>After Lunch</th><th>Afternoon</th></tr>
<tr><td rowspan="3">Monday
June 17th
Tutorials</td><td colspan="2">MaRTE OS: Bringing Embedded Systems and Real-Time POSIX Together,
M. González and M. Aldea</td><td colspan="2">Using Open Source Hardware and Software to Build Reliable Systems,
J. Sherrill and J. Gaisler</td></tr>
<tr><td colspan="2">Principles of Physical Software Design in Ada95,
M. Heaney</td><td colspan="2">Implementing Design Patterns in Ada95,
M. Heaney</td></tr>
<tr><td colspan="4">SPARK, an "Intensive Overview",
P. Amey and R. Chapman</td></tr>
<tr><td rowspan="2">Tuesday
June 18th
Sessions &
Exhibition</td><td rowspan="2">Embedded Systems unsuitable for object orientation,
Maarten Boasson</td><td>Embedded Systems</td><td>Real-Time Systems</td><td rowspan="2">High Integrity Systems</td></tr>
<tr><td>Case Studies</td><td>Vendor presentations</td></tr>
<tr><td rowspan="2">Wednesday
June 19th
Sessions &
Exhibition</td><td rowspan="2">On Architectural Stability and Evolution,
Mehdi Jazayeri</td><td rowspan="2">Ada Language Issues</td><td>Program Analysis</td><td rowspan="2">Tools</td></tr>
<tr><td>Vendor presentations</td></tr>
<tr><td rowspan="2">Thursday
June 20th
Sessions &
Exhibition</td><td rowspan="2">Reasoning About Reliable Distributed Programs,
Rachid Guerraoui</td><td>Distributed Systems</td><td>Libraries</td><td rowspan="2">Contextware: Bridging Physical and Virtual Worlds,
Alois Ferscha</td></tr>
<tr><td>Vendor presentations</td><td>OO Technology</td></tr>
<tr><td rowspan="3">Friday
June 21st
Tutorials &
Workshop</td><td colspan="4">Improved Software Testing with the Use of Metrics,
A. Sorkowitz</td></tr>
<tr><td colspan="2">WG9 Meeting</td><td colspan="2">CORBA 3 and CORBA for Embedded Systems,
S. Ron Oliver</td></tr>
<tr><td colspan="2">Workshop: A Standard Container Library for Ada,
E. Lamm</td><td colspan="2">Cleanroom Software Engineering: An Overview,
W. Bail</td></tr>
</table>

Invited Speakers

Embedded Systems Unsuitable for Object Orientation

Maarten Boasson, Quaerendo Invenietis bv & University of Amsterdam

It will be argued that the current focus on object technology is detrimental to progress in embedded systems. The core of the problem is that OO is fine for analysis but does not answer the design needs. Solutions for shortcomings are sought within the OO dogma, making things worse. This talk will outline a different approach.

Maarten Boasson studied mathematics in Groningen, the Netherlands. He became involved in advanced studies aiming at control of complexity, both of the development process and of the system under development itself. This resulted in the creation of a novel architecture for distributed reactive systems, that has been applied successfully in numerous systems and is, more than 10 years after its introduction, still unsurpassed in its support for integration, fault tolerance and component reuse. In 1996 Boasson was appointed professor of computer science at the University of Amsterdam, where he holds a chair in Industrial Complex Computer Systems. He played a major role in establishing a dutch national research program in embedded systems, and is currently associate editor-in-chief of IEEE Software.

Reasoning About Reliable Distributed Programs

Rachid Guerraoui, Swiss Federal Institute of Technology in Lausanne (EPFL)

What does it mean for a distributed program to be reliable? A program is reliable if it looks like a centralized program that does never fail. This talk aims at addressing the ramifications underlying this first glance intuitive answer. While doing so, the talk overviews several decades of work on correctness of distributed programs, from Lamport's atomicity and Papadimitrious' serializability, to linearizability and x-ability.

Rachid Guerraoui is professor in computer science at the Swiss Federal Institute of Technology in Lausanne (EPFL). He leads the Distributed Programming Laboratory and teaches object-oriented programming and distributed algorithms. He is interested in devising abstractions for reliable distributed programming.

Contextware: Bridging Physical and Virtual Worlds

Alois Ferscha, University of Linz

Alois Ferscha joined the University of Linz as full professor in 2000. He published more than 60 technical papers on topics related to parallel and distributed computing. Currently his research interests are in the areas of Pervasive Computing, Embedded Software Systems, Wireless Communication, Multiuser Cooperation, Distributed Interaction and Distributed Interactive Simulation.

On Architectural Stability and Evolution

Mehdi Jazayeri, Technical University of Vienna

Many organizations are now pursuing software architecture as a way to control their software development and evolution challenge. A software architecture describes the properties of a family of products, thus addressing the problems of both development and evolution. An important problem is to be able to evaluate the "goodness" of a proposed architecture. The talk will propose stability or resilience as a measure of goodness of an architecture. The stability of an architecture is a measure of how well it accommodates new family members. It can be measured by the amount of code changes necessary for the introduction of a new member. A case study of several releases of a telecommunication software system containing a few million lines of code will be used to demonstrate one way to try to estimate architectural stability. The talk will also present the challenges in software evolution and conclude with recommendations for future research.

Mehdi Jazayeri is a professor of computer science at the Technical University of Vienna. He spent many years in software research and development at several Silicon Valley companies, including ten years at Hewlett-Packard Laboratories in Palo Alto, California. His recent work has been concerned with component-base software engineering of distributed systems, particularly Web-based systems. He is a coauthor of Programming Language Concepts (John Wiley, 1998), Fundamentals of Software Engineering (Prentice-Hall, 2002), and Software Architecture for Product Families (Addison-Wesley, 2000).

Other Program Details

Exhibiting

Exhibition space will be provided at the Parkhotel Schönbrunn in the area of the so-called "Kaisersalon". The exhibition and a summary of the exhibits will be publicized in handouts, conference schedule, and conference program. Announcements will be made in the course of technical presentations.

Sponsoring

A sliding scale of sponsorship provides a range of benefits. All levels include display of the sponsor's logo on the conference web site and it the program.

Social Program

Several activities have already been organized. On Tuesday the City of Vienna has invited us all for a reception at the historic town hall. Before that we will enjoy a guided tour by bus that will provide a first impression of the city and several of its well-known sights.

Wednesday evening the conference banquet will take place at a famous "Heurigen" in Grinzing. Over a glass of wine and traditional Viennese cuisine we will have the opportunity to experience several of the mundane ingredients such as "Schrammel-Musik" and "Wiener Gemütlichkeit" that add to the flair of this city.

See the conference web site for more details (http://www.ada-europe.org/conference2002.html).

Organization

Conference Chair

Gerhard H. Schildt
Technical University Vienna
Department of Computer-Aided Automation
Schildt@auto.tuwien.ac.at

Program Co-Chairs

Johann Blieberger
Technical University Vienna
Department of Computer-Aided Automation
Blieberger@auto.tuwien.ac.at

Alfred Strohmeier
Swiss Fed. Inst. of Technology Lausanne
Software Engineering Lab
Alfred.Strohmeier@epfl.ch

Tutorial Chair

Helge Hagenauer
University of Salzburg
Dept. Comp. Science & System Analysis
hagenau@cosy.sbg.ac.at

Exhibition Chair

Thomas Gruber
Austrian Research Centers Seibersdorf
thomas.gruber@arcs.ac.at

Publicity Chair

Dirk Craeynest
Offis nv/sa & K.U.Leuven
Dirk.Craeynest@cs.kuleuven.ac.be

Local Organization Chair

Bernd Burgstaller
Technical University Vienna
Department of Computer-Aided Automation
Burgstaller@auto.tuwien.ac.at

In cooperation with

SIGAda

With the support of

The city of Vienna

http://www.ada-europe.org/conference2002.html

The SPARK way to Correctness is Via Abstraction

John Barnes

11 Albert Road, Caversham, Reading RG4 7AN, United Kingdom; Tel +44 118 947 4125

Abstract

This paper gives a short introduction to the SPARK language and illustrates how the use of abstraction leads towards correctness.

Keywords: Abstraction, Spark, Ada

Introduction

Abstraction is a key concept in the design of many systems whether they be made of intangible software or real hard stuff such as an automobile. A good system will be such that the various components interact through well-defined interfaces in an appropriate manner. This should eliminate unwanted interactions which might occur if the interfaces are not properly defined. The brake pedal of your car should not change the volume of the radio and so on. This desirable state can be achieved by ensuring that interactions only occur via defined interfaces and moreover that the functionality of the components are completely and correctly specified by the interface definitions (the whole truth and nothing but the truth).

Ada provides interfaces through specifications – typically package specifications containing subprogram specifications. However, these subprogram specifications do not provide a full definition of the subprograms. All they provide is enough information to enable the compiler to construct calls of the subprograms but say little if anything about what the subprograms might actually do. Although the Ada approach enables information hiding to be achieved and good component specifications to be written, and indeed encourages these through its style, nevertheless it does not ensure correctness and completeness.

SPARK enables Ada specifications to be strengthened by providing more information about interfaces and the behaviour of components. This extra information can be provided at various levels. At the simplest level it ensures that a component can only interact with certain objects but need say nothing about what it does to them; at the highest level it provides a complete definition of what it does to the objects. At the simplest level it thus prevents unexpected side effects whereas at the highest level it can lead to complete proofs of correctness.

SPARK should be looked upon as a language in its own right. In practical terms, it is a subset of Ada with additional information provided through annotations which take the form of Ada comments. Programs are therefore compiled with a normal Ada compiler and in addition are examined with independent SPARK tools which also analyse the annotations.

It is often felt that formal tools are hard to use and require a great deal of effort. One of the advantages of SPARK is its flexibility. It can be used for formal proof but a great deal of benefit can be obtained by its use at the simplest level which requires little effort. This paper outlines some important features of SPARK using a number of examples.

Abstraction

The first part of this paper introduces the basic ideas of abstraction and refinement.

A simple example

We start by considering a very simple example which shows how the SPARK annotations increase the level of information concerning abstraction. Consider the information given by the following Ada procedure specification

```
procedure Add(X: in Integer);
```

Frankly, it tells us very little. It just says that there is a procedure called Add and that it takes a single parameter of type Integer whose formal name is X. But it says nothing about what the procedure does. It might do anything at all. It certainly doesn't have to add anything nor does it have to use the value of X. It could for example subtract two unrelated global variables and print the result to some file. But now consider what happens when we add the lowest level of SPARK annotation. The specification might become

```
procedure Add(X: in Integer);
--# global in out Total;
```

This states that the only global variable that the procedure can access is that called Total. Moreover it has mode information similar to that of parameters; indeed a global variable can be looked upon as a parameter in which the actual is always the same. The SPARK rules also say more about the modes. Whereas in Ada the modes provide permission to read or update as appropriate, in SPARK such reading or updating is mandatory (SPARK generally abhors unused entities). So the specification tells us that the initial value of Total must be used (**in**) and that a new value will be produced (**out**) and also that the parameter X (**in**) must be used.

So now we know rather a lot. We know that a call of Add will produce a new value of Total and that it will use the initial value of Total and the value of X. We also know that Add cannot affect anything else. It certainly cannot print anything nor have any other malevolent side effect.

The next level of annotation gives the detailed dependency relations so that the specification becomes

```
procedure Add(X: in Integer);
--# global in out Total;
--# derives Total from Total, X;
```

In this particularly simple example, this adds no further information. We already knew that we had to use X and the initial value of Total and produce a new value of Total and this is precisely what this derives annotation says.

Finally we can add the third level of annotation which concerns proof and obtain

```
procedure Add(X: in Integer);
--# global in out Total;
--# derives Total from Total, X;
--# post Total = Total~ + X;
```

The postcondition explicitly says that the final value of Total is the result of adding its initial value (distinguished by ~) to the value of X. So now the specification is complete.

It is important to emphasize that these annotations are part of the procedure specification. (In the case of distinct specification and body, the annotations are not repeated in the body; if there is no distinct specification then they occur in the body before the reserved word **is**.) The annotations separate the interaction between the caller and the specification from that between the specification and the implementation. Hence the Examiner (the main SPARK tool) carries out two sets of checks; it checks that the annotations are consistent with the procedure body and it also checks that the annotations are consistent with each call of the procedure.

Thus when we come to implement Add, if we access a global other than Total or use Total or X in a way inconsistent with the mode information then the SPARK Examiner will produce appropriate error messages.

Generally, the higher levels of annotation enable the Examiner to carry out a more searching analysis.

State

The idea of state is vitally important. Programs do things by changing the state of objects in a general sense. In Ada, state is typically held in the form of variables in packages. A simple example is provided by a random number generator in which the state of the sequence is held in a variable hidden in a package body. Consider

```
package Random_Numbers
--# own Seed;
--# initializes Seed;
is
  procedure Random(X: out Float);
  --# global in out Seed;
  --# derives X, Seed from Seed;

end Random_Numbers;

package body Random_Numbers is
  Seed: Integer;
  Seed_Max: constant Integer := ... ;

  procedure Random(X: out Float) is
  begin
    Seed := ... ;
    X := Float(Seed) / Float(Seed_Max);
  end Random;

begin          -- initialization part
  Seed := 12345;
end Random_Numbers;
```

This example shows the package body containing the declaration of a variable Seed and the body of the subprogram Random. Each call of Random updates the value of Seed using some pseudo-random algorithm and then updates X by dividing by the constant Seed_Max. Each successive value of Seed depends upon the previous value and is preserved between calls of Random. The variable Seed is initialized in the initialization part of the package body.

This example also illustrates a number of other annotations. The variable Seed has to be mentioned in both an own annotation and an initialization annotation of the package specification. The own annotation makes it visible to other annotations and the initializes annotation indicates that it must be initialized by the elaboration of the package. The procedure Random contains a global annotation for Seed as well as a derives annotation.

The initializes annotation can also be satisfied by initializing Seed in its declaration. An alternative approach might be to declare some procedure Start in the package Random_Numbers (to be called from outside) whose purpose is to assign a first value to Seed. In this case an initializes annotation would not be required but the Examiner will complain if flow analysis reveals that Random is being called before Start.

It is important to observe that from the Ada point of view the variable Seed is not declared until the body and is thus not known to the compiler at the point of the specification of the subprogram Random. However, Seed is a global variable of Random from the point of view of SPARK and thus must be mentioned in the annotation for Random so that flow through Random may be tracked; the own annotation ensures that Seed is known to the Examiner at the specification of Random.

The derives annotation shows explicitly that each call of Random produces a number X derived from Seed and also modifies Seed. As mentioned earlier this annotation is optional.

The variable Seed is protected from manipulation by users of the procedure Random by being declared within the body of the package although it is visible in the annotations in the specification. It could be argued that making the existence of Seed known to the user is a violation of abstraction. However, we certainly ought to know that the procedure Random does something to some state external to itself otherwise we could deduce that each call of Random would inevitably produce the same value each time it is called. On the other hand we don't need to know

exactly what Seed is and indeed in this example the external view reveals no details.

Abstract state machines

The random number package is a very simple example of an abstract state machine. In general an abstract state machine is an entity, which has well defined states plus a set of operations, which cause state transitions; properties of the state can be observed by calling appropriate functions.

An abstract state machine is typically represented in Ada by a package, with variables which record its state declared in its body. Procedures that act on the machine and functions that observe its state are specified in the visible part of the package specification. All other details are hidden in the package body.

The following shows the full details of a single stack treated as an abstract state machine with the state initialized automatically on elaboration.

```
package The_Stack
--# own S, Pointer;
--# initializes Pointer;
is
  procedure Push(X: in Integer);
  --# global in out S, Pointer;
  --# derives S from S, Pointer, X &
  --#             Pointer from Pointer;

  procedure Pop(X: out Integer);
  --# global in S; in out Pointer;
  --# derives Pointer from Pointer &
  --#             X from S, Pointer;
end The_Stack;

package body The_Stack is
  Stack_Size: constant := 100;
  type Pointer_Range is range 0 .. Stack_Size;
  subtype Index_Range is
                Pointer_Range range 1 .. Stack_Size;
  type Vector is array (Index_Range) of Integer;
  S: Vector;
  Pointer: Pointer_Range;

  procedure Push(X: in Integer) is
  begin
    Pointer := Pointer + 1;
    S(Pointer) := X;
  end Push;

  procedure Pop(X: out Integer) is
  begin
    X := S(Pointer);
    Pointer := Pointer - 1;
  end Pop;
begin
  Pointer := 0;
end The_Stack;
```

The stack state variables S and Pointer are declared in the body of the package and Pointer is initialized. These internal variables are not directly accessible to users of the stack object. However, their existence and the existence of the initialization of Pointer are made visible to the Examiner for the purpose of analysis by the **own** and **initializes** annotations in the package specification just as the variable Seed of the package Random was made visible.

However, the above technique is not satisfactory since we have made visible considerable detail of the internal representation of the state of the machine, namely the existence of the individual variables S and Pointer. If at some later stage we need to change the implementation then there is a high risk that the specification will need to be changed because of the SPARK rules even though it would not need to be changed by the Ada rules. This would in turn give rise to tiresome dependencies since it would require all the calls to be reexamined and recompiled.

(A minor problem with the package as written is that when we come to use it we will get messages saying that S is being used before it is given a value. Of course we know that the dynamic behaviour is such that the initialization of S is unnecessary but the Examiner is not aware of this. Perhaps the best solution is simply to initialize S as well.)

Refinement

The problems of unnecessary dependencies can be overcome by using abstract own variables to provide what is known as refinement. An abstract own variable does not correspond to a concrete Ada variable at all but instead represents a set of variables used in the implementation.

As a consequence, an abstract own variable occurs in two annotations, the own variable clause in the package specification and then also in a refinement definition in the body giving the set onto which it is mapped.

The stack example could then be rewritten as

```
package The_Stack
--# own State;              -- abstract variable
--# initializes State;
is
  procedure Push(X: in Integer);
  --# global in out State;
  --# derives State from State, X;

  procedure Pop(X: out Integer);
  --# global in out State;
  --# derives State, X from State;
end The_Stack;

package body The_Stack
--# own State is S, Pointer;       -- refinement definition
is
  Stack_Size: constant := 100;
  type Pointer_Range is range 0 .. Stack_Size;
  subtype Index_Range is
                Pointer_Range range 1 .. Stack_Size;
  type Vector is array (Index_Range) of Integer;
  S: Vector;
  Pointer: Pointer_Range;
```

```
   procedure Push(X: in Integer)
   --# global in out S, Pointer;
   --# derives S from S, Pointer, X &
   --#             Pointer from Pointer;
   is
   begin
     Pointer := Pointer + 1;
     S(Pointer) := X;
   end Push;

   procedure Pop(X: out Integer)
   --# global in S; in out Pointer;
   --# derives Pointer from Pointer &
   --#             X from S, Pointer;
   is
   begin
     X := S(Pointer);
     Pointer := Pointer - 1;
   end Pop;

begin                          -- initialization
   Pointer := 0;
   S := Vector'(Index_Range => 0);
end The_Stack;
```

This enables the more abstract specification to be linked with the concrete body. The refinement acts as the link and says that the abstract own variable State is implemented by the two concrete variables S and Pointer.

Note moreover that the subprogram bodies have to have a refined version of their global and derives annotations (if provided) written in terms of the concrete variables.

One consequence of the refinement is that both Pointer and S have to be initialized because we have promised that the abstract variable State will be initialized. Of course, as mentioned earlier, we know that the dynamic behaviour is such that the initialization of S is unnecessary and we could omit it in practice and ignore the consequential message from the Examiner.

The various constituents of the refinement must either be variables declared immediately within the package body (such as S and Pointer) or they could be own variables of private child packages or of embedded packages declared immediately within the body. The process of refinement can be repeated since an own variable in the constituent list might itself be an abstract own variable of the child or embedded package.

It is worth summarizing some key points regarding the visibility of state variables of abstract state machines.

- The **own** annotation of an abstract state machine makes the existence of its state visible wherever the machine is visible.
- Annotations of subprograms external to a machine which (indirectly) read or update its state (by executing subprograms of the machine) must indicate that they import or export the machine state.
- Only the existence of the machine state (and its reading or updating) is significant in this context. The details can still be hidden by refinement.

The second point is important and states that annotations have to be explicitly transitive. Thus a procedure that calls Push and Pop also has to be annotated to indicate that it changes the state of the stack.

```
procedure Use_Stack
--# global in out The_Stack.State;
--# derives The_Stack.State from The_Stack.State;
is
begin
  The_Stack.Push( ... );
  ...
  The_Stack.Pop( ... );
  ...
end Use_Stack;
```

Finally note that one abstract state machine could be implemented using another abstract state machine embedded within it. Thus if a machine B is to be embedded in a machine A, this can be done by embedding the package representing B in the body of the package representing A. The state of B can then be represented as an item in the refinement. Alternatively the package representing B could be a private child of the package representing A.

Refinement of course relates to top-down design and provides a natural way of implementing such a design. It is especially important that refinement can be cascaded; this avoids a combinatorial explosion of visible data items which might otherwise occur especially in large programs. The key point is that it makes the existence of state known without giving away the details - the irrelevant detail is kept hidden.

The location of state

It is very important to ensure that state is located sensibly. In order to illustrate this first consider the following simple example

```
procedure Exchange(X, Y: in out Float)
--# derives X from Y &
--#             Y from X;
is
  T: Float;
begin
  T := X;  X := Y;  Y := T;
end Exchange;
```

The parameters X and Y have mode **in out**. This requires them to be both read and updated. The (optional) derives annotation in addition states that the final value of X depends upon the initial value of Y and vice versa. Note that the final value of X does not depend upon the initial value of X.

The scope of program objects should always be as restricted as possible. The rules of SPARK discourage the use of a global variable simply as a ‘temporary store’. For

example we might try to redefine the procedure Exchange so that the temporary T is global by writing

```
procedure Exchange(X, Y: in out Float)
--# global out T;
--# derives X from Y &
--#             Y from X;
is
begin
  T := X;  X := Y;  Y := T;
end Exchange;
```

But this is illegal because it violates one of several rules of completeness. The one that is violated here is that every variable mentioned in a global definition must be used somewhere in the dependency relation. We have to add T to the derives annotation thus

```
--# derives X from Y &
--#             Y, T from X;
```

and this forces us to admit that we actually change T. Moreover, flow analysis of a call of Exchange will reveal the use of T. Thus a succession of calls such as

```
Exchange(A, B);
Exchange(P, Q);
```

results in the following message from the Examiner

```
              Exchange(A, B);
              ^1
!!! (  1)   Flow Error    : Assignment to T is
            ineffective.
```

This is because the value of T produced by the first call of Exchange is overwritten by the second call without being used. Remember that analysis of the calls is done using only the abstract view presented by the specification and so the internal use of the value of T in the body is not relevant. Note further that this message will be produced even if the optional derives annotation is omitted.

Unnecessary state should thus be avoided. Indeed, the use of unnecessary state as in this example requires annotations for T on the subprogram calling Exchange and so on transitively. The annotations therefore cascade and so the use of unnecessary state is very painful and thereby discouraged.

But some state is necessary and we have seen how refinement may be used to ensure that although the existence of state in an abstract state machine must be made visible, nevertheless the fine details are properly hidden. (We can have our abstraction cake and still eat it!)

There is an interesting analogy between abstraction through refinement and the composition of records out of components. Consider a private type defining a position where the full type reveals the details in terms of *x*- and *y*-coordinates

```
type Position is private;

...

type Position is
  record
    X_Coord, Y_Coord: Float;
  end record;
```

Such a record type is sensible because the two coordinates are logically related; we can then consider a value of the type Position as a single entity which can be manipulated as a whole without knowing the details of its inner construction.

Refinement allows an abstract own variable to provide an external view of a more detailed set of variables within the package. Using the analogy to records, we should only use refinement to group together naturally related items. Thus the refinement of the variable State of the package The_Stack into the variables Pointer and S is appropriate.

Proof

For some applications formal proof is a valuable technique for showing correctness. SPARK has comprehensive facilities for proof including the ability to develop proofs with refinement when there are two views of a state. In order to illustrate this it is necessary to explain some of the basic techniques involved.

The proof process

The general idea is that we state certain hypotheses which we assert are always satisfied when a subprogram is called (the *preconditions*) and we also state the conditions which we want to be satisfied as a result of the call (the *postconditions*). These conditions are given as further annotations in the subprogram specification. We then have to show that the postconditions always follow from the preconditions.

The Examiner processes the text and generates one or more theorems (conjectures really since they might not turn out to be true) which then have to be proved in order to show that the postconditions do indeed always follow from the preconditions. These theorems which are called *verification conditions* are often trivially obvious. If they are not then there are two tools which can be used. These are the Simplifier which carries out routine simplification and the Proof Checker which is an interactive assistant that enables the user to explore the problem and hopefully construct a valid proof.

In order for the proof tools to function correctly, they need to be aware of the various rules which can be used. For the predefined types these are built into the system but other rules can be provided as we shall see in a moment.

As a first example consider once more the procedure Exchange. There is no precondition since it is designed to work no matter what the values of the parameters happen to be. But there is of course a postcondition and so the procedure becomes

```
procedure Exchange(X, Y: in out Float)
--# derives X from Y &
--#             Y from X;
--# post X = Y~ and Y = X~ ;
```

```
is
  T: Float;
begin
  T := X;  X := Y;  Y := T;
end Exchange;
```

Note again the use of the tilde character with in out parameters; the decorated form indicates the initial imported value of the parameter whereas the undecorated form indicates the final exported value.

The verification condition generated by the Examiner for the procedure Exchange is

```
H1:   true .
      ->
C1:   y = y .
C2:   x = x .
```

The notation used is that there are a number of hypotheses (H1, H2, ...) followed by a number of conclusions (C1, C2, ...) which have to be verified using the hypotheses. Note that the conditions are written in a language known as FDL (Functional Definition Language) which has a strong mathematical flavour.

In this example there is no precondition and so effectively no hypotheses (this is represented as the single hypothesis H1 which is true). The two conclusions to be proved are that y = y and x = x which are reasonably self-evident and so it is pretty clear that the procedure Exchange is correct.

If we were stubborn and wanted to be completely confident then we could submit the above verification condition to the Simplifier which would reduce it to simply

```
*** true .        /* all conclusions proved */
```

Verification conditions often appear mysterious and not obviously related to the code; they are produced by a "hoisting process" whereby the postcondition is transformed backwards through the statements in order to arrive at the so-called weakest precondition; this is the condition that must hold at the start in order for the postcondition to hold. We then have to show that the weakest precondition follows from the given precondition. In the verification condition, the hypotheses correspond to the given precondition and the conclusions to be proved correspond to the weakest precondition. However, the details of the hoisting transformations need not concern us in this paper.

Loops

Significant computations usually have loops and these cause complexity in proving correctness. The problems arise because the code of a loop is usually traversed a number of times with different conditions.

The approach taken is to cut a loop so that the various parts can be treated separately. The cut is made by inserting an assert statement which gives conditions that are to be true at that point. The conditions can be thought of as postconditions for the sequence of code arriving at the cutpoint and as preconditions for the sequence going on from the cutpoint.

A simple example is provided by the following integer division algorithm which might be used on a processor without a hardware divide instruction.

```
procedure Divide(M, N: in Integer; Q, R: out Integer)
--# derives Q, R from M, N;
--# pre (M >= 0) and (N > 0);
--# post (M = Q * N + R) and (R < N) and (R >= 0);
is
begin
  Q := 0;
  R := M;
  loop
    --# assert (M = Q * N + R) and (R >= 0);
    exit when R < N;
    Q := Q + 1;
    R := R - N;
  end loop;
end Divide;
```

Each transversal of the loop adds one to the trial quotient and subtracts the divisor N from the corresponding trial remainder until the remainder first becomes less than the divisor. Clearly it only works if both M and N are not negative and also the divisor must not be 0; hence the precondition.

The postcondition has two parts. First the output parameters must have the appropriate mathematical relation implied by the division process and secondly the remainder must be less than the divisor and not negative, so we have

```
--# post (M = Q * N + R) and (R < N) and (R >= 0);
```

The choice of assertion is fairly obvious. As noted above, the final postcondition has two parts, the division relation and the upper and lower bounds on the remainder. All the loop does is keep the division relation true and reduce the remainder until it satisfies the upper bound (as well as keeping the lower bound satisfied). The assertion is simply that the division relation is true and that the remainder satisfies the lower bound; the exit statement is taken when the upper bound is satisfied as well. The initial statements before the loop are designed to ensure that the assertion is true when the loop is first entered.

There are therefore three sections of code to be verified. They are from the start to the beginning of the loop, around the loop, and from the loop to the end. The assert statement acts as the postcondition for the first section and as the precondition for the last section. It also acts as both precondition and postcondition for the loop itself; since it is unchanged by the loop it is often referred to as a loop invariant.

When the Examiner is applied to this subprogram, it produces verification conditions corresponding to the three sections. From the start to the assertion the verification condition is

```
H1:   m >= 0 .
H2:   n > 0 .
      ->
C1:   m = 0 * n + m .
C2:   m >= 0 .
```

Conclusion C2 is trivially obvious since it is just the hypothesis H1. Conclusion C1 is pretty obvious as well.

The verification condition for going around the loop from assertion to assertion is

```
H1:   m = q * n + r .
H2:   r >= 0 .
H3:   not (r < n) .
      ->
C1:   m = (q + 1) * n + (r - n) .
C2:   r - n >= 0 .
```

and that from the assertion to the final end is

```
H1:   m = q * n + r .
H2:   r >= 0 .
H3:   r < n .
      ->
C1:   m = q * n + r .
C2:   r < n .
C3:   r >= 0 .
```

In all cases the Simplifier reduces all the conclusions to true. It is also quite straightforward to show that they are true by hand – although perhaps a little tedious in the case of the loop itself which requires some manipulation. However, such trivial manipulation is prone to error if done by hand and the great advantage of the Simplifier is that it does not make careless mistakes.

Having shown that the verification conditions for the three separate sections of code are true it then follows that the procedure is correct. (To be honest we have only proved that it is partially correct; this means that it is correct provided that it terminates.)

In practice one does not bother to look at the unsimplified conditions and so the process is quite straightforward.

Proof functions

Annotations such as postconditions can be very expressive. Not only can we use the variables of the program but various other notations are also available. We have already noted the use of the tilde character to distinguish initial and final values of in out parameters. The following examples illustrate other possibilities.

```
type Atype is array (Index) of T;

procedure Swap_Elements(I, J: in Index;
                        A: in out Atype);
--# derives A from A, I, J;
--# post A = A~[I => A~(J); J => A~(I)];
```

The postcondition means that the final value of A is the initial value with elements I and J interchanged. Note carefully that it is the initial value of A that is referred to on the right hand side and so there are three uses of the tilde character.

```
function Max(X, Y: Integer) return Integer;
--# return M => (X >= Y -> M = X) and
--#           (Y >= X -> M = Y);
```

This illustrates that functions have return annotations rather than postconditions. The annotation should be read as return M such that if X >= Y then M is X and if Y >= X then M is Y.

```
function Value_Present(A: Atype; X: T) return Boolean;
--# return for some M in Index => (A(M) = X);
```

This function returns true if at least one component of the array has the value X. Remember that Index is the index type of the array type Atype.

```
function Find(A: Atype; X: T) return Index;
--# pre Value_Present(A, X);
--# return Z => (A(Z)) = X) and
--#       (for all M in Index range Index'First .. Z-1 =>
--#           (A(M) /= X));
```

This function returns the index of the first component of the array with the value X. Note the precondition which uses the previous function to ensure that such a value does exist. All Ada functions can be used in annotations in this way with any global variables being added as explicit additional parameters (remember the earlier remark that global variables can be looked upon as parameters that are always the same).

Sometimes, however, the functional nature of the annotation language is not rich enough in which case we can add our own so-called proof functions which do not exist as Ada functions at all.

As an elementary example consider the following implementation of the factorial function

```
--# function Fact(N: Natural) return Natural;

function Factorial(N: Natural) return Natural
--# pre N >= 0;
--# return Fact(N);
is
  Result: Natural := 1;
begin
  for Term in Integer range 1 .. N loop
    Result := Result * Term;
    --# assert Term > 0 and Result = Fact(Term);
  end loop;
  return Result;
end Factorial;
```

The approach we take is to introduce a proof function Fact which we can use in the annotations even though it is not defined in the Ada program text. An interesting observation is that although recursion is not permitted in SPARK because dynamic storage is forbidden, nevertheless proof rules can use recursion in their definition because proof is done offline independently of program execution.

The Examiner is now able to produce verification conditions; it does this without needing to know what the proof function Fact actually means because the process of

producing verification conditions simply involves formal substitution.

There are four paths including one from start to finish which bypasses the loop in the case of N being zero. We will look at the verification conditions for just two of them. That from the assertion to the finish is

```
H1:   term > 0 .
H2:   result = fact(term) .
H3:   term = n .
      ->
C1:   result = fact(n) .
```

This is clearly correct by simply substituting from H3 into H2 irrespective of what Fact actually means. That from assertion to assertion is more interesting

```
H1:   term > 0 .
H2:   result = fact(term) .
H3:   not (term = n) .
      ->
C1:   term + 1 > 0 .
C2:   result * (term + 1) = fact(term + 1) .
```

In order to prove this we need a mathematical theorem for the Fact function namely

$$fact(n) = n \times fact(n\text{-}1) \quad n > 0$$

The other two paths need the other obvious mathematical theorem

$$fact(0) = 1$$

In order to prove the verification conditions using the Proof Checker, it is necessary to give the Checker the rules corresponding to the above theorems. These can be expressed in the following form

```
rule_family fact:
  fact(X) requires [X : i] .

fact(1): fact(N) may_be_replaced_by
                              N * fact(N-1) if [N > 0] .
fact(2): fact(0) may_be_replaced_by 1 .
```

Given such rules the proofs can be entirely mechanized.

The reader might feel that this is all a bit of a cheat. However, the approach is typical of many safety-related mechanisms. Two routes to the solution are provided using entirely different technologies; one uses the Ada program and the other uses the annotations and proof rules. Since they agree we have a high degree of confidence in their correctness.

Proof and refinement

We are now in a position to return to the theme of abstraction and consider how we might add annotations for proof to the stack example.

We saw how we could have two views of the state of the package The_Stack – an external abstract view provided by the abstract variable State and an internal concrete view provided by the two variables S and Pointer. In order to develop proofs we need to map abstract conditions for the external view onto concrete conditions for the internal view. The package might become

```
package The_Stack
--# own State: Stack_Type;        -- abstract variable
--# initializes State;
is
  --# type Stack_Type is abstract;          -- proof type

  --# function Not_Full(S: Stack_Type) return Boolean;
  --# function Not_Empty(S: Stack_Type)
                                          return Boolean;
  --# function Append(S: Stack_Type; X: Integer)
                                       return Stack_Type;

  procedure Push(X: in Integer);
  --# global in out State;
  --# pre Not_Full(State);
  --# post State = Append(State~, X);

  ...   -- similarly Pop

end The_Stack;

package body The_Stack
--# own State is S, Pointer;        -- refinement definition
is
  ... -- etc as before

  procedure Push(X: in Integer)
  --# global in out S, Pointer;
  --# pre Pointer < Stack_Size;
  --# post Pointer = Pointer~ + 1 and
  --#         S = S~[Pointer => X];
  is
  begin
    Pointer := Pointer + 1;
    S(Pointer) := X;
  end Push;

  ...   -- similarly Pop plus initialization

end The_Stack;
```

The above omits the derives annotation partly for simplicity but also to emphasize that derives annotations are not necessary in order to develop proofs although we have shown them in earlier examples for completeness.

The abstract own variable State now includes a type announcement for the proof type Stack_Type. In developing the verification conditions, the Examiner converts this proof type into an FDL record type having two components corresponding to the variables S and Pointer. (Note again the strong analogy between refinement and record composition.)

There are also proof functions Not_Full and Append (with parameters of the proof type) which are used to give the pre- and postconditions for Push. The proof function Not_Empty is required for Pop.

Three verification conditions are generated for Push – one shows that the refined precondition follows from the abstract precondition, one shows that the abstract postcondition follows from the refined postcondition and

the other (the usual one) shows that the refined postcondition follows from the refined precondition. The first is

```
H1:   not_full(state) .
H2:   s = fld_s(state) .
H3:   pointer = fld_pointer(state) .
      ->
C1:   pointer < stack_size .
```

The notation should be self-evident, H2 means that the refined variable S corresponds to the field s of the abstract State.

To complete the proofs we need proof rules for the proof functions in terms of the concrete variables such as

```
not_full(S) may_be_replaced_by
                          fld_pointer(S) < stack_size .
```

Given such rules the verification conditions can all be proved.

The stack package might be used by external procedures which themselves have proof annotations in terms of the proof functions. Of course they can only see the external view of the stack and so rules need to be developed in terms of that view. But the rules can themselves be proved using the concrete view.

Design and implementation

One of the goals of this paper is to show that SPARK uses abstraction as a key ingredient in showing correctness. The important thing about abstraction is controlling the level of visibility. We are familiar in Ada with the idea of having more than one view of a type, for example the full view and the partial view of a private type. SPARK allows private types of course but as we have seen extends this idea of views to the representation of state through refinement. We have also seen how proofs may be developed around the two representations.

But it must not be thought that proof is the major goal of SPARK. The real goal is developing correct programs more cheaply and also of course convincing the customer that they are correct within a given budget. Sometimes formal proof is the appropriate tool to being convinced that the program is correct – but for most purposes it would be overkill.

But perhaps the real strength of SPARK is that it encourages good design by revealing the flow of information. For example, suppose we have a package Stuff which contains a procedure Do_It which in turn calls the procedures Push and Pop and thereby manipulates The_Stack. The Ada structure might be

```
package Stuff is
  procedure Do_It;
end Stuff;

with The_Stack;
package body Stuff is
  procedure Do_It is
  begin
    ...
    The_Stack.Push( ... );
    ...
    The_Stack.Pop( ... );
    ...
  end Do_It;
end Stuff;

with Stuff;
procedure Main is
begin
  Stuff.Do_It;
end Main;
```

By just looking at the procedure Main we have absolutely no idea what it does. Even if we look at the specification of Stuff we are none the wiser. We have to look at the body of Stuff to see that it has access to The_Stack. Clearly this is against the spirit of separation of specification and body. The specification ought to tell us what something does whereas the body should simply tell us how it does it. Of course the very fine detail is not always relevant but at least we ought to be clear about what is affected by looking at the specification.

Now consider the same example with the minimal SPARK annotations.

```
--# inherit The_Stack;
package Stuff is
  procedure Do_It;
  --# global in out The_Stack.State;
end Stuff;

with The_Stack;
package body Stuff is
  procedure Do_It is
  begin
    ...
    The_Stack.Push( ... );
    ...
    The_Stack.Pop( ... );
    ...
  end Do_It;
end Stuff;

with Stuff;
--# inherit The_Stack, Stuff;
--# main_program;
procedure Main
--# global in out The_Stack.State;
is
begin
  Stuff.Do_It;
end Main;
```

This introduces two more annotations. One is the inherit clause which is required on the specification of a package in order to give access to other packages. The other is the main program annotation. The global annotations now reveal that the state of the package The_Stack is being manipulated by the procedure Do_It and (transitively) by the main subprogram. The fine details of just what is being

done to The_Stack are not revealed and indeed it is probably not necessary to know at this structural level.

But the key point is that the side effect of manipulating the state of the stack is revealed. The annotations encourage good design because a bad design will often have a lot of curious unexpected side effects which are embarrassingly revealed by the annotations. Changing the structure in order to reduce the complexity of annotations will simplify the design by increasing coherence and reducing unnecessary cross-coupling.

Design relates to the specifications of components and their interrelationships whereas implementation relates to their bodies. It is interesting to note that most SPARK annotations apply to specifications and this emphasizes that SPARK is primarily about encouraging good design which then in turn leads to correctness of implementation.

An important issue is scalability, that is the ability to cope with large programs as well as small ones. In this context it is important that refinement can be cascaded. Thus if a component C uses a subcomponent S such as the stack as implementation detail then this fact need not be revealed at the top level. The subcomponent S can be embedded in C or (equivalently) be a private child of C. The state of C can then be refined to include the state of S so that S becomes just an implementation detail.

Note carefully that the most benefit will be obtained from SPARK if it is used as early as possible in the design process. It can weed out poor design before energy is spent on implementation. Of course, SPARK is valuable at the implementation stage as well because it will statically detect many errors that the compiler cannot detect. Indeed, SPARK reaches parts of the program that other tools do not reach.

Levels of use

One of the beauties of SPARK is that it can be used at different levels according to the requirements of the project. The simplest level just requires visibility annotations such as global and own annotations. These alone enable the Examiner to detect a great many errors that cannot be found by the compiler and thus have to be found by the tedious process known as testing often at a later stage in the development process and thus both more expensive to find and to fix.

We know that a key strength of Ada is its strong typing which reveals errors that in a pathetic language such as C have to be found by testing. SPARK extends this capability of Ada by finding even more errors without testing.

At the lowest level of annotation, flow analysis detects many typical errors such as uninitialized variables (those read before being given a value), ineffective parameters (whose value has no effect on the outcome), overwritten values (values that are overwritten before being used), nonterminating loops, aliasing, and so on. In addition many of the errors that can be made in Ada (such as inadvertently using the wrong variable because a later declaration hides it) cannot occur in SPARK because of stricter naming rules.

The introduction of the derives annotation will give more detail of the interactions between components and analysis will then often reveal surprising cross-coupling indicative of poor design or coding errors.

Proof may be appropriate for algorithmic applications. Proof can be applied at several levels as well. This paper has described proof whereby the user is required to add proof annotations. Another option is to check for the absence of runtime errors such as those that arise from violating a bound of an array. Since the Examiner knows about the type model it can generate verification conditions which show the absence of such runtime errors without the user having to supply any annotations at all. Proof can be performed with or without the derives annotations so in fact there are really many levels at which SPARK can be used.

These different levels can be mixed up within a single program. The computational leaves of a system might be subject to proof, the derives annotation might be useful for intermediate subcomponents whereas the outermost part of the system might well have the lowest level of annotation. This is a big strength of SPARK; it can be seen as several tools rolled into one each appropriate to a different part of a project.

Conclusion

Abstraction has been the main theme of this paper. Good abstraction is about revealing relevant detail and hiding irrelevant detail. Plain Ada programs typically do not reveal all the relevant detail. But SPARK with its refinement capability can be used to reveal the detail that matters while keeping the irrelevant detail hidden.

The reader should be aware that this paper has only surveyed some of the capabilities of SPARK. Much has been omitted such as how to interface to external parts of a system. Further details will be found in [1] from which many of the examples given here have been taken and which includes a CD containing demonstration versions of the SPARK tools plus full documentation.

Finally it should be noted that SPARK is well-established and has been successfully used on many projects in a variety of application areas; see for example [2, 3].

References

[1] J. G. P. Barnes (1997), *High Integrity Ada - The SPARK Approach*, Addison-Wesley.

[2] R. C. Chapman (2000), *Industrial Experience with SPARK*, Proceedings of SIGAda 2000.

[3] M. Croxford and J. Sutton (1996), *Breaking Through the V and V Bottleneck*, Proceedings of Ada in Europe Conference 1995, Lecture Notes in Computer Science 1031, Springer-Verlag.

Ada-Europe Associate Members (National Ada Organizations)

Ada-Belgium

attn. Dirk Craeynest
c/o Offis nv/sa
Weiveldlaan 41/B32
B-1930 Zaventem
Belgium

Phone: +32-2-725-40-25
Fax: +32-2-725-40-12
Email: Dirk.Craeynest@offis.be
URL: www.cs.kuleuven.ac.be/~dirk/ada-belgium

Ada in Denmark

attn. Jorgen Bundgaard
c/o DDC-I
Gl. Lundtoftevej 1B
DK-2800 Lyngby
Denmark

Phone: +45-45-871144
Fax: +45-45-872217
Email: jb@ddci.dk

Ada-Deutschland

attn. Dr. Peter Dencker
Aonix GmbH
Durlacher Allee 95
D-76137 Karlsruhe
Deutschland

Phone: +49-721-98653-22
Fax: +49-721-98653-98
Email: dencker@aonix.de
URL: ada-deutschland.de

Ada-France

chez Fabrice Kordon
48 rue Vergniaud
F-75013 Paris
France

Phone: +33-1-44 27 61 89
Fax: +33-1-44 27 62 86
Email: bureau@ada-france.org
URL: www.ada-france.org

Ada-Spain

attn. Francisco Perez-Zarza
P.O. Box 50.403
E-28080 Madrid
Spain

Phone: +34-1-627-8247
Fax: +34-1-309-3685
Email: fperez@ceselsa.es
URL: www.adaspain.org

Ada in Sweden

Ada I Sverige
c/o Mariadata
Box 1085
SE-141 22 Huddinge
Sweden

Phone: +46-08-779-88-30
Fax: +46-08-774-37-93
Email: info@ada-i-sverige.se
URL: www.ada-i-sverige.se

Ada in Switzerland

attn. Alfred Strohmeier
Software Engineering Laboratory
Swiss Federal Institute of Technology Lausanne
CH-1015 Lausanne EPFL
Switzerland

Phone: +41 21 693 4231
Fax +41 21 693 5079
Email: alfred.strohmeier@epfl.ch
URL: http://lglwww.epfl.ch/Ada-in-Switzerland

Ada Language UK

attn. Helen Byard
P.O. Box 322
York YO10 3GY
UK

Phone: +44-(0)1904-412740
Fax: +44-(0)1904-426702
Email: admin@adauk.org.uk
URL: www.adauk.org.uk

Ada UK 2001 Sponsors

ACT Europe
Contact: Franco Gasperoni
8, Rue de Milan, 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

Alenia Marconi Systems
Contact: Don Harvey
Eastwood House, Glebe Rd., Chelmsford, Essex, CM1 1QW, UK
Tel: +44-(0)1276-696901 Fax: +44-(0)1276-659842
Email: don.harvey@amsjv.com URL: www.aleniamarconisystems.com

Aonix Europe Ltd
Contact: Neil Michniak
Partridge House, Newtown Rd., Henley on Thames, Oxon, RG9 1HG, UK
Tel: +44-(0)14941-415000 Fax: +44-(0)14941-571866
Email info@aonix.co.uk URL: www.aonix.com

ARTiSAN Software Tools
Contact: Peter Kibble
Stamford House, Regent St., Cheltenham, Glos., GL50 1HN, UK
Tel: +44-(0)1242-229320 Fax: +44-(0)1242-229301
Email: peterk@artisansw.com URL: www.artisansw.com

BAE SYSTEMS
Contact: Paul McCormack
Warwick House, PO Box 87, Farnborough Aerospace Centre, Farnborough, Hants, GU14 6YU, UK
Email: Paul.McCormack@baesystems.com URL: www.baesystems.com

Data Systems and Solutions
Contact: Dave Woodhall
SEAS Building, Sinfin Lane, Derby, DE24 8BJ, UK
Tel: +44-(0)1332-771700 Fax: +44-(0)1332-770921
Email: info@ds-s.com URL: www.ds-s.com

EDS
Contact: Lee Edwards
Hartley House, 15 Bartley Wood Business Park, Bartley Way, Hook, Hants., RG27 9XA, UK
Tel: +44-(0)1256-741122 Fax: +44-(0)1256-741132
Email: swep.sales@eds.com

First Matrix Ltd
Contact: Alan Barker
Old Lion Court, High St,. Marlborough, Wilts., SN8 1HQ., UK
Tel: +44-(0)1672-515510 Fax: +44-(0)1672-515514
Email: arb@ftmx.com

Green Hills Software Ltd
Contact: Jon Williams
Chancery Court, Lincoln Rd., High Wycombe, Bucks., HP12 3RE., UK
Tel: +44-(0)1844-267950 Fax: +44-(0)1844-267955
Email: sales-uk@ghs.com URL: www.ghs.com

IPL Information Processing Ltd
Contact: Ian Gilchrist
Eveleigh House, Grove St., Bath, BA1 5R., UK
Tel: +44-(0)1225-475114 Fax: +44-(0)1225-444400
Email: ipl@iplbath.com URL: www.iplbath.com

LDRA Ltd
Contact: Jim Kelly
24 Newtown Rd., Newbury, Berks., RG14 7BN, UK
Tel: +44-(0)635-528828 Fax: +44-(0)635-528657
Email: sales@ldra.com URL: www.ldra.com

Objektum
Contact: Derek Russell or Ahmed Amin
Units 2/3 Cranleigh Works, The Common, Cranleigh, GU6 8SB, UK
Tel: +44-(0)1483-278178 Fax: +44-(0)1483-275384
Email: info@objektum.com URL: www.objektum.com

Praxis Critical Systems Ltd
Contact: Peter Amey
20 Manvers St., Bath, BA1 1PX, UK
Tel: +44-(0)1225-469991 Fax: +44-(0)1225-469006
Email: sparkinfo@praxis-cs.co.uk URL: www.praxis-cs.co.uk

Rational Software Ltd
Contact: Roger Bowser
Kingswood, Kings Ride, Ascot, Berks., SL5 8AJ, UK
Tel: +44-(0)1344-295000 Fax: +44-(0)1344-295001
Email: info@rational.com URL: www.rational.com

John Robinson & Associates
Contact: John Robinson
2 Currer St., Oakenshaw, Bradford, W. Yorks., BD12 7DP, UK
Tel: +44-(0)1274-691935 Fax: +44-(0)8700-558750
Email: John@jr-and-assoc.demon.co.uk URL: www.jr-and-assoc.demon.co.uk

Telelogic UK Ltd
Contact:
Chancery House, 8 Edward St., Birmingham, B1 2RX, UK
Tel: +44-(0)121-2346600 Fax: +44-(0)121-2346611
Email: info@telelogic.com URL: www.telelogic.com

TNI Europe Ltd
Contact: Tony Elliston
58a Mill St., Congleton, Cheshire, CW12 1AG, UK
Tel: +44-(0)1260-291449 Fax: +44-(0)1260-291449
Email: info@tni-europe.com URL: www.tni-europe.com

Wind River Systems UK Ltd
Contact: David Bew
Unit 5 & 6, 1st Floor, Ashted Lock Way, Aston Science Park, Birmingham, B7 4AZ, UK
Tel: +44-(0)121-3590999 Fax: +44-(0)121-3804444
Email: inquiries-uk@windriver.com URL: www.windriver.com