ADA USER JOURNAL

Volume 44
Number 4
December 2023

Contents

Quarterly News Digest

Alejandro R. Mosteo

Centro Universitario de la Defensa de Zaragoza, 50090, Zaragoza, Spain; Instituto de Investigación en Ingeniería de Aragón, Mariano Esquillor s/n, 50018, Zaragoza, Spain; email: amosteo@unizar.es

Contents

[Messages without subject/newsgroups are replies from the same thread. Messages may have been edited for minor proofreading fixes. Quotations are trimmed where deemed too broad. Sender's signatures are omitted as a general rule. —arm]

Preface by the News Editor

Dear Reader,

One of the most significant features of Ada 2022 is the new light-weight parallelism. Although no Ada compiler implements it as of this writing, equivalent features are now available in library form thanks to Tucker Taft [1]. Let us hope this brings us near an actual Ada 2022 implementation.

While preparing this issue, I learned about the "Beaujolais Effect" [2], a challenge issued by Ada's original designer, Jean Ichbiah. He offered a bottle of Beaujolais wine to the person that found an example of changing the behavior of an Ada 83 program by adding/removing a "use" clause. Was there ever a winner? Find out at the referenced thread!

Sincerely,

Alejandro R. Mosteo.

[1] "Light-weight Parallelism Threading Library Based on Ada 2022 Features", in Ada-related Tools.

[2] "Beaujolais Challenge", in Ada Practice.

Ada-related Events

Ada Monthly Meetup 2023

From: Fernando Oleo / Irvise <irvise_ml@irvise.xyz>
Subject: Re: Ada Monthly Meetup 2023
Date: Wed, 11 Oct 2023 19:16:21 +0200
Newsgroups: comp.lang.ada

[Past events for the record. —arm]

I would like to announce the November Ada Monthly Meetup which will be taking place on the 4th of November at **14:00 UTC time (15:00 CET)**. As always the meetup will take place over at Jitsi. Hopefully this time I will not have the same amount of technical issues…

If someone would like to propose a talk or a topic, feel free to do so! Streaksu, the creator of the [Ironclad] (https://ironclad.cx/) kernel, has volunteered to give an introductory talk and demonstration of the OS :)

Here are the connection details from previous posts:

The meetup will take place over at Jitsi, a conferencing software that runs on any modern browser. The link is [Jitsi Meet] (https://meet.jit.si/AdaMonthlyMeetup) The room name is "AdaMonthlyMeetup" and in case it asks for a password, it will be set to "AdaRules".

I do not want to set up a password, but in case it is needed, it will be the one above without the quotes. The room name is generally not needed as the link should take you directly there, but I want to write it down just in case someone needs it.

Best regards and see you soon!
Fer

P.S: careful with the time! In the EU we will enter daylight savings time at the end of October. For that reason, I have decided to change the meeting to UTC 14:00h! Please, check your local timezones!

P.P.S: the October meeting went pretty well! We had quite a few people and two presentations, one from Francesc, who introduced [Alice] (https://github.com/alice-adventures/Alice) and Rod Kay, who showed his work on [SWIG4Ada] (https://github.com/charlie5/swig4ada).

The meeting was livestreamed to Youtube and can be watched [here] (https://www.youtube.com/watch?v=0Pnuy663gZM) (the video resolution is only 360p for the time being).

From: Fernando Oleo / Irvise <irvise_ml@irvise.xyz>
Date: Mon, 13 Nov 2023 22:19:27 +0100

I would like to announce the December Ada Monthly Meetup which will be taking place on the 2nd of December at **14:00 UTC time (15:00 CET)**. As always the meetup will take place over at Jitsi. The Meetup will also be livestreamed to Youtube :)

If someone would like to propose a talk or a topic, feel free to do so! We currently have no topics ;)

Here are the connection details from previous posts: [...]

2nd Call for Contributions - AEiC 2024

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Subject: Ada-Europe Conference - 2nd Call for Contributions - AEiC 2024
Date: Wed, 22 Nov 2023 16:54:55 -0000
Newsgroups: comp.lang.ada, fr.comp.lang.ada, comp.lang.misc

[CfC is included in the Forthcoming Events Section —arm]

Grants for Open Access - AEiC 2024

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Subject: AEiC 2024 - Ada-Europe conference - grants for Open Access
Date: Thu, 21 Dec 2023 17:14:52 -0000
Newsgroups: comp.lang.ada, fr.comp.lang.ada

Season's greetings from the organizers of the 28th Ada-Europe International Conference on Reliable Software Technologies (AEiC 2024), to be held 11-14 June 2024, in Barcelona, Spain!

Accepted Journal Track papers will be published in the conference's Special Issue of the Journal of Systems Architecture (JSA). Note that the Ada-Europe organization will waive the Open Access fees for the first four accepted papers, which do not already enjoy OA from other agreements with the Publisher.

www.ada-europe.org/conference2024/cfp.html#cfpjournal
(V3.1)

Ada-related Resources

[Delta counts are from October 10th to February 19th. —arm]

Ada on Social Media

From: Alejandro R. Mosteo <amosteo@unizar.es>
Subject: Ada on Social Media
Date: 19 Feb 2024 16:03 CET
To: Ada User Journal readership

Ada groups on various social media:

- Reddit: 8_561 (+139) members [1]
- LinkedIn: 3_479 (+25) members [2]
- Stack Overflow: 2_393 (+28) questions [3]
- Gitter: 243 (+14) people [4]
- Telegram: 173 (+15) users [5]
- Ada-lang.io: 182 (+36) users [6]
- Libera.Chat: 76 (+4) concurrent users [7]

[1] http://old.reddit.com/r/ada/

[2] https://www.linkedin.com/groups/114211/

[3] http://stackoverflow.com/questions/tagged/ada

[4] https://app.gitter.im/#/room/#ada-lang_Lobby:gitter.im

[5] https://t.me/ada_lang

[6] https://forum.ada-lang.io/u

[7] https://netsplit.de/channels/details.php?room=%23ada&net=Libera.Chat

Repositories of Open Source Software

From: Alejandro R. Mosteo <amosteo@unizar.es>
Subject: Repositories of Open Source software
Date: 19 Feb 2024 16:10 CET
To: Ada User Journal readership

GitHub: 1000* (=) developers [1]

Rosetta Code: 940 (=) examples [2]

38 (=) developers [3]

Alire: 393 (+23) crates [4]

Sourceforge: 248 (+1) projects [5]

Open Hub: 214 (=) projects [6]

Codelabs: 57 (=) repositories [7]

Bitbucket: 37 (=) repositories [8]

* This number is an unreliable lower bound due to GitHub search limitations.

[1] https://github.com/search?q=language%3AAda&type=Users

[2] https://rosettacode.org/wiki/Category:Ada

[3] https://rosettacode.org/wiki/Category:Ada_User

[4] https://alire.ada.dev/crates.html

[5] https://sourceforge.net/directory/language:ada/

[6] https://www.openhub.net/tags?names=ada

[7] https://git.codelabs.ch/?a=project_index

[8] https://bitbucket.org/repo/all?name=ada&language=ada

Language Popularity Rankings

From: Alejandro R. Mosteo <amosteo@unizar.es>
Subject: Ada in language popularity rankings
Date: 19 Feb 2024 16:13 CET
To: Ada User Journal readership

[Positive ranking changes mean to go up in the ranking. —arm]

- TIOBE Index: 25 (-2) 0.77% (=) [1]
- PYPL Index: 15 (+1) 1.08% (+0.04%) [2]
- Stack Overflow Survey: 42 (=) 0.77% (=) [3]
- IEEE Spectrum (general): 36 (=) Score: 0.0107 (=) [4]
- IEEE Spectrum (jobs): 29 (=) Score: 0.0173 (=) [4]
- IEEE Spectrum (trending): 30 (=) Score: 0.0122 (=) [4]

[1] https://www.tiobe.com/tiobe-index/

[2] http://pypl.github.io/PYPL.html

[3] https://survey.stackoverflow.co/2023/

[4] https://spectrum.ieee.org/top-programming-languages/

Ada-related Tools

UXStrings 0.6.0

From: Blady <p.p11@orange.fr>
Subject: [ANN] Release of UXStrings 0.6.0
Date: Sat, 14 Oct 2023 18:33:19 +0200
Newsgroups: comp.lang.ada

This Ada library provides Unicode character strings of dynamic length. It is now available on Alire [1] in version 0.6.0.

Changes:

- Add string convenient subprograms [2]: Contains, Ends_With, Starts_With, Is_Lower, Is_Upper, Is_Basic, Is_Empty, Remove, Replace.
- Add list of strings with convenient subprograms [3]: Append_Unique, Filter, Join, Remove_Duplicates, Replace, Slice, Sort, Is_Sorted, Merge and Split on strings.

So far in UXStrings, its API are similar to those of the strings Ada standard libraries. If you find some missing, make your proposals on Github.

NB: UXStrings3 is now the default implementation.

[1] https://alire.ada.dev/crates/uxstrings.html

[2] https://github.com/Blady-Com/UXStrings/blob/master/src/uxstrings3.ads#L346

[3] https://github.com/Blady-Com/UXStrings/blob/master/src/uxstrings-lists.ads

[4] https://github.com/Blady-Com/UXStrings/issues

Source Code for the ARM Formatting Tool

From: Vincent D. <vincent.diemunsch@gmail.com>
Subject: Source code for the ARM Formatting Tool
Date: Wed, 25 Oct 2023 14:15:00 -0700
Newsgroups: comp.lang.ada

I have tried to download the source code of the formatting tool from the site http://ada-auth.org/arm.html, but it seems that the package ARM_Paragraph is missing.

Does anyone know how to get this file?

From: Maxim Reznik <reznikmm@gmail.com>
Date: Thu, 26 Oct 2023 01:15:19 -0700

I have a github repository synced with ada-auth Web CVS. I was able to build the formatting tool from the source.

https://github.com/reznikmm/ada-auth/

From: Vincent D. <vincent.diemunsch@gmail.com>
Date: Thu, 26 Oct 2023 07:58:13 -0700

Thank you for the link on GitHub, but the build didn't work for me: I get the same error regarding "ARM_Paragraph" package missing.

```
$ git clone https://github.com/reznikmm/ada-auth.git
$ cd ada-auth
$ gprbuild -p -P ada_form.gpr
Setup
[mkdir] object directory for project Ada_Form
Compile
[Ada] arm_form.ada
arm_form.ada:6:06: error: file "arm_paragraph.ads" not found
arm_form.ada:6:06: error: "Arm_Formatter
```

(body)" depends on "Arm_Master (spec)"
arm_form.ada:6:06: error: "Arm_Master (spec)" depends on "Arm_Format (spec)"
arm_form.ada:6:06: error: "Arm_Format (spec)" depends on "Arm_Paragraph (spec)"
gprbuild: *** compilation phase failed

From: Vincent D. <vincent.diemunsch@gmail.com>
Date: Thu, 26 Oct 2023 08:46:04 -0700

Versions that compile:

- 4500f560 Corrected note format for ISO version
- 29db0326 Split out the normative references clause.
- ff3db3ca Various updates for FDIS work and draft 34.

Versions that do not compile:

- 0e95e912 Various updates for FDIS 2.0.
- 4d93b18c A number of small formatting changes, mostly only for the FDIS.
- 260566bd Various updates for FDIS/Draft 35.

The problem appears in version 0e95e912 "Various updates for FDIS 2.0." from the 23/09/2022 where with ARM_Paragraph is added to arm_frm.adb but the package was not added to the sources.

From: Simon Wright <simon@pushface.org>
Date: Thu, 26 Oct 2023 18:06:19 +0100

> the package was not added to the sources.

Because it's not in CVS.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Tue, 31 Oct 2023 20:56:54 -0500

> Because it's not in CVS.

It is now. And it always was in the ZIP file of the source. (That is made from the files that I use to compile the tool, so it should always be compilable.)

One of the big downsides of working at home is that some support functions get delayed until one gets into the office -- and that means that they're easily forgotten. As in this case, checking the new files into the CVS (the existing files were updated, of course, leaving a mess for anyone trying to build from the CVS).

Sorry about that.

From: Vincent D. <vincent.diemunsch@gmail.com>
Date: Fri, 3 Nov 2023 15:02:02 -0700

> It is now. And it always was in the ZIP file of the source. (That is made from the files that I use to compile the tool, so it should always be compilable.)

I am sorry, but even if I put the package "arm_paragraph" in the source code, I get compile errors. For instance in version 0e95e9125e066ce564fe369221821452535b6260:

gprbuild -p -P ada_form.gpr
Compile
[Ada] arm_form.ada
[Ada] arm_cont.adb
[Ada] arm_frm.adb
arm_frm.adb:1163:13: error: missing case value: "Usage"
arm_frm.adb:9357:53: error: "Numbered_T_and_D_List" not declared in "ARM_Database"
arm_frm.adb:9362:53: error: "T_and_D_List" not declared in "ARM_Database"
arm_frms.adb:1415:33: error: unmatched actual "Note1_Text" in call
gprbuild: *** compilation phase failed

And with the latest: gprbuild -p -P ada_form.gpr

Compile
[Ada] arm_form.ada
arm_form.ada:263:41: error: "Rest" not declared in "ARM_Master"
gprbuild: *** compilation phase failed

What is the ZIP file of the source that you mentioned? Where can I find it?

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 16 Nov 2023 19:17:08 -0600

On each of the individual Reference Manual pages (that is, Ada 2012, Ada 2022, etc.) on Ada-Auth.org, at the bottom, you will find links for the formatting tool, one for a Windows executable, one for the CVS, and one for a ZIP file containing the source.

I'm not sure why you are seeing compilation problems with the CVS; it appears complete and correct on my end. Did you make sure that you have the latest versions of all of the files (they were updated on October 3rd)?

For example, the ARM_Frm.Adb file should have a change entry of: - 9/11/23 - RLB - Added Usage category and commands. and of course have code for the Usage category and commands. I've started some work for the post-Ada 2022 RM (currently known as Ada 202y) - the tool is constantly evolving.

One of the advantages of using the ZIP files is that they reflect the tool as it was used to generate a specific version of the RM; the "current" version of the tool probably only has been tested on the "current" version of the RM source and thus it is not certain to work perfectly.

Light-weight Parallelism Threading Library Based on Ada 2022 Features

From: Tucker Taft <tucker.taft@gmail.com>
Subject: Light-weight parallelism threading library based on Ada 2022 features
Date: Tue, 31 Oct 2023 15:43:44 -0700
Newsgroups: comp.lang.ada

A full implementation of the parallel features of Ada 2022 is yet to be released. In the meantime, here is a light-weight-threading library that provides essentially all of the parallel features of Ada 2022, using various generics, etc. Scheduling is provided using a plug-in architecture. If no scheduler is plugged in, the light-weight threads are simply executed sequentially. If a light-weight-thread scheduler is plugged in, then the light-weight threads spawned by instances of the various generic packages are managed by that scheduler.

There are currently two LWT scheduler plug-ins:

* a wrapper for the GNU implementation of OpenMP (lwt-openmp.ads)
* a work-stealing based plug-in, written entirely in Ada

Below is a link to the "readme.md" documentation for the GitHub lwt library. It is currently part of the ParaSail GitHub repository, but the files in "lwt" are actually independent of ParaSail. ParaSail has its own work-stealing-based scheduler built-in, but at some point we plan to shift over to using the "lwt" library. But at the moment, there is no dependence either way between the ParaSail interpreter/compiler and the lwt library.

https://github.com/parasail-lang/parasail/tree/main/lwt#light-weight-threading-library-for-ada-2022

Feel free to open GitHub Issues if you find problems with the implementation, or have suggestions for improvements.

Enjoy!

-Tucker Taft

The ParaSail GitHub repository was created by my colleague Olivier Henley, and he has also helped to improve the documentation and testing scripts. Much appreciated!

GtkAda Contributions 3.32

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Subject: ANN: GtkAda contributions v3.32
Date: Mon, 18 Dec 2023 12:34:59 +0100
Newsgroups: comp.lang.ada

The library is an extension of GtkAda dealing with the following issues:

- Tasking support;
- Custom models for tree view widget;
- Custom cell renderers for tree view widget;
- Multi-columned derived model;
- Extension derived model (to add columns to an existing model);
- Abstract caching model for directory-like data;

- Tree view and list view widgets for navigational browsing of abstract caching models;
- File system navigation widgets with wildcard filtering;
- Resource styles;
- Capturing resources of a widget;
- Embeddable images;
- Some missing subprograms and bug fixes;
- Measurement unit selection widget and dialogs;
- Improved hue-luminance-saturation color model;
- Simplified image buttons and buttons customizable by style properties;
- Controlled Ada types for GTK+ strong and weak references;
- Simplified means to create lists of strings;
- Spawning processes synchronously and asynchronously with pipes;
- Capturing asynchronous process standard I/O by Ada tasks and by text buffers;
- Source view widget support;
- SVG images support.

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

Changes (18 December 2023) to the version 3.31:

- Get_CSS_Name and Set_CSS_Name were added to Gtk.Missed;
- Gtk.Widget.Styles.CSS_Store changed to enumerate labels of Gtk_Notebook or its descendants;
- Since Gtk broke its CSS rules and the widget class cannot be used in the CSS style anymore and because unset widget names are defaulted to the class name but have no effect on the style name, Gtk.Widget.Styles.CSS_Store was changed to report only the widget name if different from the class names.

Ada and Operating Systems

Spurious Error with GNAT 13.2.0 on Intel MacOS 17.2

From: Moi <findlaybill@blueyonder.co.uk>
Subject: spurious error with GNAT 13.2.0 on Intel macOS 17.2
Date: Thu, 14 Dec 2023 23:49:15 +0000
Newsgroups: comp.lang.ada

The 17.2 update is accompanied by updated Command Line Tools, so, having made a copy of the current CLTs, I let it update.

With GNAT 13.2.0 on Intel macOS 17.2 this happens:

/Users/wf/KDF9/emulation/Testing: chmod 444 ST0

On compiling and running the minimum test case:

```
with Ada.Direct_IO;
with Ada.IO_Exceptions;
procedure failure is
 package my_IO is new
    Ada.Direct_IO(Integer);
  use my_IO;
     my_file : File_Type;
begin
   Open(my_file, Inout_File, "ST0");
exception
   when Ada.IO_Exceptions.Use_Error =>
     raise program_error with "Open
     failed to get RW access";
end failure;
```

I get:

A. with ”-largs -Wl,-ld_classic” in the linker parameters:

/Users/wf/KDF9/emulation/Testing: failure raised PROGRAM_ERROR: Open failed to get RW access

This is what should happen.

B. recompiled and relinked without ”-largs -Wl,-ld_classic”:

/Users/wf/KDF9/emulation/Testing: failure raised CONSTRAINT_ERROR: erroneous memory access

OOPS!

Strangely, this is now the ONLY one out of dozens of regression tests that fails in this software configuration. Previously, they all failed at the link stage.

From: Simon Wright <simon@pushface.org>
Date: Fri, 15 Dec 2023 15:23:31 +0000

> The 17.2 update is accompanied by updated Command Line Tools, so, having made a copy of the current CLTs, I let it update.

I think you mean 14.2!

GCC 13/CLT 15.* still has issues with exceptions, eg this unresolved issue: https://github.com/iains/gcc-13-branch/issues/10, and the ld-classic dance fixes them as far as I can tell.

I'm working with this fix: https://github.com/simonjwright/xcode_15_fix

which is included in my GCC 13.2.0 build: https://github.com/simonjwright/distributing-gcc/releases/tag/gcc-13.2.0-aarch64

> Strangely, this is now the ONLY one out of dozens of regression tests that fails in this software configuration. Previously, they all failed at the link stage.

Yes, the 15.1 betas fixed the linking problem. I've no idea why some exceptions don't get caught/raise memory access errors. I do think that the aarch64 GCC 14 has fixed the problem.

Ada Inside

First Ada DO-178 Certification

From: Jeffrey R.Carter <spam.jrcarter.not@spam.acm.org.not>
Subject: First Ada DO-178 Certification
Date: Wed, 4 Oct 2023 13:39:39 +0200
Newsgroups: comp.lang.ada

Does anyone remember when Ada avionics S/W was first certified to DO-178? My memory is 1980s, but I haven’t been able to find any information about it.

Ada and Other Languages

Upcasting Interfaces with CPP Convention in GNAT

From: Kura <kuraitou@gmail.com>
Subject: Upcasting interfaces with CPP convention in GNAT
Date: Thu, 2 Nov 2023 05:20:08 -0700
Newsgroups: comp.lang.ada

I'm trying to figure out how to correctly perform an upcast in GNAT using interface types that have the CPP convention. I have two types corresponding to C++ classes: IBase and IDerived, each with corresponding access types suffixed with _Ptr. The library that I'm wrapping returns a pointer to some concrete implementation of IDerived, but the moment I convert it to an IBase_Ptr to pass it to functions within the library, the program segfaults while performing some kind of tag check. I've found information on this topic to be very sparse, and the GNAT manual has no discussion about allowed conversions or examples showing how to do this. This seems like it should be an allowed conversion and the program works as expected if I either replace the usage of Base_Ptr with Derived_Ptr within the function wrapper specification (not feasible because there are many types deriving from Base_Ptr and adding overloads would affect its vtable) or use Unchecked_Conversion between the two pointer types (unsure if this is safe). I'm also aware I could lay out the vtables manually and do away with tagged types, but I would like to avoid that if possible.

Here is a minimal example that segfaults, all compiled with the same toolchain on windows/mingw64:

==== repro.adb

```
with Wrap; use Wrap;
procedure Repro is
   P : IDerived_Ptr := GetDerivedInstance;
   Q : IBase_Ptr := IBase_Ptr (P);
begin
  null;
end Repro;
```

==== wrap.ads

```
package Wrap is
   type IBase is interface;
   pragma Convention (C_Plus_Plus, IBase);
   type IBase_Ptr is access all IBase'Class;

   type IDerived is interface and IBase;
   pragma Convention (C_Plus_Plus,
   IDerived);
type IDerived_Ptr is access all
IDerived'Class;

   function GetDerivedInstance return
   IDerived_Ptr with Import => True,
   Convention => C_Plus_Plus,
   External_Name => "GetDerivedInstance";
end Wrap;
```

==== lib.cpp

```
class IBase {};
class IDerived : public IBase {};
class Impl : public IDerived {
public:
        Impl() = default;
};

extern "C"

IDerived* GetDerivedInstance() {
        return new Impl();
}
```

==== end example

Error and stack trace:

```
Thread 1 received signal SIGSEGV,
Segmentation fault.
ada.tags.offset_to_top
(this=(system.address) 0x6591710) at a-
tags.adb:806
806 a-tags.adb: No such file or directory.
(gdb) bt
#0 ada.tags.offset_to_top
(this=(system.address) 0x6591710) at
a-tags.adb:806
#1 ada.tags.base_address
(this=(system.address) 0x6591710) at
a-tags.adb:286
#2 ada.tags.displace (this=(system.address)
0x6591710, t=0x7ff7fe8301c8
<wrap__ibaseT+8> (wrap.ibase)) at
a-tags.adb:354
#3 0x00007ff7fe82649c in repro () at
C:\repro\src\repro.adb:5
```

Any ideas?

From: J-P. Rosen <rosen@adalog.fr>
Date: Thu, 2 Nov 2023 15:11:54 +0100

I don't know if this is the cause of your problem, but you should give convention C_Plus_Plus to the pointer types too (IBase_Ptr and IDerived_Ptr).

From: Kura <kuraitou@gmail.com>
Date: Thu, 2 Nov 2023 08:08:24 -0700

That did not solve my problem, but thank you for the tip.

From: Kura <kuraitou@gmail.com>
Date: Fri, 3 Nov 2023 14:13:05 -0700

I was able to work around the issue by using abstract tagged null records instead of interfaces - no other changes necessary. It seems that interfaces can only be at the very top of a hierarchy even if you're only extending another interface.

Ada vs. Rust for Low-level System Software

From: Nasser M. Abbasi <nma@12000.org>
Subject: Ada vs. Rust for low level system software
Date: Tue, 12 Dec 2023 22:28:40 -0600
Newsgroups: comp.lang.ada

Has anyone made a study of differences between Rust and Ada for low level hardware system software?

Since Ada is mainly used in this area, why has Rust, which is a much younger language, and target this same area has gained so much popularity but not Ada?

https://dl.acm.org/doi/fullHtml/10.1145/3551349.3559494

"Rust is a rising programming language designed to build system software [4, 10, 20]. On the one hand, Rust offers access to and control of the low-level system resources. On the other hand, unlike conventional systems programming languages, Rust ensures memory and concurrency safety"

"Rust often inserts bound checks at the execution time to rule out out-of-bound accesses"

Well, does not Ada also "ensures memory and concurrency safety" and checks for out-of-bound accesses?

I am just wondering what Rust brings to the table that Ada does not have and why is Rust becoming so popular when Ada is not.

I never used Rust myself, but used Ada.

Has anyone done a study comparing the two languages or knows both that can give some comments on this?

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Wed, 13 Dec 2023 09:27:34 +0100

> Has anyone made study [...]

What for? Any language comparisons lost their meaning long ago as the whole language business degraded into hobbyist/corporate bullshit.

> why has Rust [...] gained so much popularity but not Ada?

Because it is always someone's arbitrary decision.

In my view Rust brings nothing and moreover is a huge step back compared to Ada. Its main and only idea is to force the programmer to explicitly manage memory through references where Ada simply uses object notation regardless of the mechanism doing the same under the hood.

Safety comes not through references but by limiting the number of cases you must resort to using dynamic allocation for statically scoped objects. E.g. Unbounded_String in Ada.

As for my major concern - the type system and the abstraction mechanisms in general, there is nothing in Rust at all.

Then of course Rust continues the worst practices tried by Ada and C++: templates/generics, macros.

From: Jeffrey R.Carter <spam.jrcarter.not@spam.acm.org.not>
Date: Wed, 13 Dec 2023 09:44:34 +0100

> Has anyone made study of difference between Rust and Ada for low level hardware system software?

You might be interested in this discussion: https://www.reddit.com/r/ada/comments/18c2nr4/where_is_ada_safer_than_rust/

From: Luke A. Guest <laguest@archeia.com>
Date: Wed, 13 Dec 2023 09:10:20 +0000

>> why has Rust gained so much popularity but not Ada?

I would say because they aimed to be a C++ replacement.

[...]

> templates/generics, macros.

What's the alternative to generics?

From: Dmitry A. Kazakov <mailbox@dmitry-kazakov.de>
Date: Wed, 13 Dec 2023 10:53:28 +0100

> What's the alternative to generics?

The question is what is the alternative to static/parametric polymorphism. The answer is dynamic polymorphism.

1. Dynamic polymorphism in Ada is as static as generics are. No run-time penalty unlikely to C++.
2. It covers cases generics do not, e.g. you can have class-wide run-time objects and proper class-wide subprograms.
3. It supports modular programming en large. E.g. you can put a class member in a dynamically linked library.
4. It is fully testable. Generics are fundamentally non-testable, only concrete instances are.

5. It does not create a meta language layer with complexities for the compiler and programmer. Advanced generic code is close to unmaintainable.

From: Kevin Chadwick <kc-usenet@chadwicks.me.uk>
Date: Mon, 18 Dec 2023 12:52:08 -0000

Ada's ranged type system coupled with its excellent record overlays make Ada a much better choice than Rust for safe and easy hardware register and network protocols.

I can't imagine that ownership for data structures on a single runtime is the reason (before SPARK got support).

A lot of Ada code will not run on any runtime but Rust also has nostd for embedded use.

They often say Rust has excellent C interfacing support but Ada's appears to be even better.

Perhaps it is ease of use and guide availability which has improved or simply perception and a lack of knowledge about Ada.

Rather than language merits, a lot of people only care about job availability, money and library availability today but might consider a risk if they perceive a future demand increase.

As to why the likes of Google and Microsoft are putting money behind it when Ada would have been a better investment. You would have to ask them. Tell me why Google continues to write security sensitive code like matter in C++? When it could be written in Ada or Rust with a C binding.

Even Javascript engines like Mozilla's spidermonkey still has so little Rust code. Though Mozilla does have more financial concerns and its competitors are already trying to say their browsers are faster. Yet Google's websites use umpteen domains slowing browsing down anyway.

Ada Practice

Get_immediate Echo Character

From: Richardthiebaud <thiebauddick2@aol.com>
Subject: get_immediate echoe character--compiled error?
Date: Sun, 1 Oct 2023 22:42:39 -0400
Newsgroups: comp.lang.ada

When I build and run the following program using GNAT 11 in Linux Mint 21.2, the keys I press are echoed on the console. According to the Ada Reference Manual, they should not be echoed. Is this a compiler error?

```
with ada.text_io; use ada.text_io;
procedure test3 is
   c: character;
   avail: boolean;
begin
   loop
      loop
         Get_Immediate(c, Avail);
         if Avail then
            exit;
         end if;
         delay 0.01;
      end loop;
   end loop;
end test3;
```

This does not happen if I call get_immediate without the avail parameter, i.e. get_immediate(c);

From: Keith Thompson <keith.s.thompson+u@gmail.com>
Date: Sun, 01 Oct 2023 22:48:36 -0700

> According to the Ada Reference Manual, they should not be echoed. Is this a compiler error?

Where does the ARM say that?

From: Richardthiebaud <thiebauddick2@aol.com>
Date: Mon, 2 Oct 2023 16:07:49 -0400

> Where does the ARM say that?

https://www.adaic.org/resources/add_content/standards/05rm/html/RM-A-10-7.html

From: Keith Thompson <keith.s.thompson+u@gmail.com>
Date: Mon, 02 Oct 2023 15:27:33 -0700

> https://www.adaic.org/resources/add_content/standards/05rm/html/RM-A-10-7.html

I don't see anything there about the character being echoed, or not.

> If a character, either control or graphic, is available from the specified File or the default input file, then the character is read; Available is True and Item contains the value of this character. If a character is not available, then Available is False and the value of Item is not specified. Mode_Error is propagated if the mode of the file is not In_File. End_Error is propagated if at the end of the file. The current column, line and page numbers for the file are not affected.

Are you assuming that not updating the current column, line, and page numbers for the file implies that the character is not echoed?

[...]

From: Richardthiebaud <thiebauddick2@aol.com>
Date: Mon, 2 Oct 2023 18:41:46 -0400

> Are you assuming that not updating the current column, line, and page numbers for the file implies that the character is not echoed?

Yes.

In any case, when it echoes the character, it increases the current column by 1, and that does contradict the Ada Reference Manual.

From: Niklas Holsti <niklas.holsti@tidorum.invalid>
Date: Tue, 3 Oct 2023 11:41:05 +0300

>> I don't see anything there about the character being echoed, or not.

Nor do I. But perhaps there should be something, since "not echoing" is useful behavior and the program can itself echo characters if that is desired.

Possibly this is why AdaCore have given different echoing behaviors to the two forms of Get_Immediate, with and without the "Available" parameter. If so, this echo difference is unfortunately coupled with the wait/no-wait behavior difference, and that coupling may be unwanted.

There are (or have been) computer terminals with local echo, where the program cannot prevent the display of each keystroke. So the "no echo" behavior cannot be an absolute requirement in the Ada manual, but it could be Implementation Advice.

> In any case, when it echos the character. it increases the current column by 1, and that does contradict the Ada Reference Manual.

You are assuming that "current column" in the Ada Reference Manual means the same as the "current column position of the terminal/screen cursor", which is not the case, so there is no formal contradiction. The Ada "current column" refers to an internal state of the file.

For an unknown type of terminal/screen, deducing the current cursor column from the stream of input characters and output characters is not feasible, because it depends on the device's interpretation of formatting control characters such as TABs and on the width of the screen or terminal window.

From: Simon Wright <simon@pushface.org>
Date: Tue, 03 Oct 2023 11:20:40 +0100

> Possibly this is why AdaCore have given different echoing behaviors [...]

The low-level Get_Immediate implementation is in sysdep.c (probably not in the adainclude/ directory in an installed compiler), in getc_immediate() and getc_immediate_nowait(), both of which call getc_immediate_common(), and I can't see any difference! ECHO gets turned off in getc_immediate_common(), regardless of caller - see link. https://github.com/gcc-mirror/gcc/blob/3ca09d684e496240a87c0327687e2898060c2363/gcc/ada/sysdep.c#L387

From: G.B.
<bauhaus@notmyhomepage.invalid>
Date: Tue, 3 Oct 2023 23:00:40 +0200

> When I build and run the following program using GNAT 11 in Linux Mint 21.2, the keys I press are echoed on the console.

Which console?

Can you try to run a C program in the same console that tests for it to be a TTY? See Simon Wright's link to GNAT's implementation. The C program would be calling isatty(0) or isatty(fileno(your_stream));

Some IDEs have a console window that is not a TTY in the sense of termios(4)/tcsetattr(3). Echoing is different, then.

From: Keith Thompson
<keith.s.thompson+u@gmail.com>
Date: Tue, 03 Oct 2023 17:13:17 -0700

> https://github.com/gcc-mirror/gcc/blob/3ca09d684e496240a87c0327687e2898060c2363/gcc/ada/sysdep.c#L387

I haven't really looked into this, but I *think* what's happening is that for the versions with the Available parameter, ECHO hasn't yet been turned off when the user types the character. If you type 'x', it echoes immediately, because the program has no way of knowing that the character will later be consumed by a call to Get_Immediate. Presumably if the user hasn't typed anything, causing Available to be set to false, Get_Immediate will turn echoing off and back on again very quickly. Echoing is disabled only for small fraction of a second it takes for Get_Immediate to be executed.

The Get_Immediate functions without the Available parameter block until a character is entered. They can disable echoing before the character is entered. Echoing will typically be disabled for minutes or seconds, from the time Get_Immediate is called and the time the user types something.

The only solution I can think of would be to disable echoing (in some non-portable manner; I don't think the standard library provides this) before the user starts typing. (Perhaps you want to run the Get_Immediate without the Available parameter in a separate task?)

From: Simon Wright
<simon@pushface.org>
Date: Wed, 04 Oct 2023 09:22:05 +0100

> I *think* what's happening is [...]

Great analysis! Is this worth raising a PR on GCC Bugzilla? (maybe only on the documentation?)

Or, alternatively, don't turn echoing off at all - what's the use case for turning it off? After all, the ARM says nothing about it.

From: Jeffrey R.Carter
<spam.jrcarter.not@spam.acm.org.not>
Date: Wed, 4 Oct 2023 12:48:41 +0200

The use case is inputting passwords and the like. See Password_Line (https://github.com/jrcarter/Encryption-utilities/blob/master/password_line.ads) for an example. Note that this has identical behavior with GNAT/Linux and ObjectAda/Windows.

From: Simon Wright
<simon@pushface.org>
Date: Wed, 04 Oct 2023 12:38:51 +0100

Obviously, you need to turn echoing off for password input. But neither the ARM nor the GNAT RM says anything about Get_Immediate's echoing behaviour, so it's hard to explain why OA does the same thing. Does its manual specify this behaviour?

From: Jeffrey R.Carter
<spam.jrcarter.not@spam.acm.org.not>
Date: Wed, 4 Oct 2023 15:05:03 +0200

Unfortunately, Ada does not provide a standard way to turn off echo.

I agree that the ARM says nothing about echo for any of its operations on Standard_Input, but clearly there is a broad consensus of Ada.Text_IO writers and users who think this is desirable behavior.

From: Niklas Holsti
<niklas.holsti@tidorum.invalid>
Date: Wed, 4 Oct 2023 19:55:51 +0300

> Great analysis!

Yes indeed.

A possible solution in Text_IO would be for Get_Immediate with Available not to enable echo when it exits. Get_Immediate with Available is typically called repeatedly, with no other input from the terminal in between these calls, so it should be ok to keep echo disabled from one such call to another. Any non-immediate input operation on the terminal (that is, on this Text_IO file) should start by re-enabling echo if it was disabled. Possibly the same should apply also to Get_Immediate without Available, that is, it should leave echo disabled, until some non-immediate input operation re-enables echo.

From: Keith Thompson
<keith.s.thompson+u@gmail.com>
Date: Wed, 04 Oct 2023 12:39:27 -0700

> A possible solution in Text_IO would be for Get_Immediate with Available not to enable echo when it exits

The *first* character typed would still echo.

I suggest that what's needed is a way to turn echoing on and off.

Meanwhile, would calling Get_Immediate *without* the Available parameter (which blocks and turns echoing off until after a character is typed) in a separate task work I haven't tried it. Of course you'd need to be careful not to have I/O calls from separate tasks interfere with each other.

From: Niklas Holsti
<niklas.holsti@tidorum.invalid>
Date: Thu, 5 Oct 2023 00:20:05 +0300

> The *first* character typed would still echo.

Only if the user is quick enough to type it before the first call of Get_Immediate.

If Get_Immediate is called for example to enter a password, usually the program will first prompt the user to "Enter password:" and then at once call Get_Immediate. Only a user who starts typing before the prompt is visible would have time to type something before the (first) call of Get_Immediate.

> I suggest that what's needed is a way to turn echoing on and off.

The user could still be quick enough to type characters before the echo is turned off, so they would echo.

> Meanwhile, would calling Get_Immediate *without* the Available parameter (which blocks and turns echoing off until after a character is typed) in a separate task work? I haven't tried it.

That should work, provided that the Ada run-time system does not block the whole program when one task blocks on an I/O request. There have been, and perhaps still are, Ada programming systems where the whole Ada program appears to the OS as a single OS thread so that one Ada task waiting on a blocking OS call blocks all other tasks in the program.

> Of course you'd need to be careful not to have I/O calls from separate tasks interfere with each other.

Yes, but other tasks should be able to output text through Standard_Output even while one task is reading Standard_Input using a blocking I/O call. Except under a one-thread run-time system.

From: Randy Brukardt
<randy@rrsoftware.com>
Date: Wed, 4 Oct 2023 19:43:55 -0500

> I agree that the ARM says nothing about echo for any of its operations on Standard_Input, but clearly there is a broad consensus of Ada.Text_IO writers and users who think this is desirable behavior.

For what it's worth, Janus/Ada turns off echoing, and that was decided without referring to any other implementation's choice in the matter. Rather, it was done to provide a way using standard calls to provide functionality that had always been available in Janus/Ada in a non-standard way.

Specifically, Janus/Ada has always had a predefined file name "KBD:" (or "/dev/kbd" on Unix), which provides raw access to the keyboard device (or standard input on more modern systems). This did not echo (or do any line editing) on CP/M and MS-DOS, and we carried that same behavior over into more modern systems.

For instance, the "Continue or Abort?" question in the compiler uses KBD: to take and discard input immediately without any waiting (usual standard input is line buffered and usually input is not processed until "enter" or similar is pressed). It seemed to us that the Get_Immediate function was intending the same sorts of uses. Note that implementing it this way makes it hard to get meaningful results if Get_Immediate is mixed with other input on the same file. (That's why we treated it as a special file in the beginning, but even that gets confused if someone else reads from Standard_Input.)

From: Keith Thompson <keith.s.thompson+u@gmail.com>
Date: Wed, 04 Oct 2023 15:12:39 -0700

>> [Initial example removed. —arm]

> I should have checked this earlier, but this does not echo with ObjectAda.

On what target system?

From: Jeffrey R.Carter <spam.jrcarter.not@spam.acm.org.not>
Date: Thu, 5 Oct 2023 11:51:15 +0200

> On what target system?

Windows.

Using Log_Float in Inline Assembler for ARM

From: Ahlan Marriott <ahlan@marriott.org>
Subject: Using Log_Float in inline assembler for ARM
Date: Sun, 19 Nov 2023 04:22:20 -0800
Newsgroups: comp.lang.ada

The following procedure Unbiased_Rounding for Float works as expected.

```
function Unbiased_Rounding (X : Float)
return Float is
  Y : Float;
begin
   Asm ("vrintn.f32 %0,%1",
   Outputs  => Float'asm_output ("=t", Y),
   Inputs   => Float'asm_input ("t", X));
   return Y;
end Unbiased_Rounding;
```

According to https://gcc.gnu.org/onlinedocs/gcc/Machine-Constraints.html the constraint t means "VFP floating-point registers s0-s31. Used for 32 bit values" and the constraint w means "VFP floating-point registers d0-d31 and the appropriate subset d0-d15 based on command line options. Used for 64 bit values only"

Therefore, we wrote our long_float version as

```
function Unbiased_Rounding
(X : Long_Float) return Long_Float is
   Y : Long_Float;
begin
    Asm ("vrintn.f64 %0,%1",
    Outputs  => Long_Float'asm_output
     ("=w", Y), Inputs   =>
     Long_Float'asm_input ("w", X));
     return Y;
 end Unbiased_Rounding;
```

However, this fails to compile. GNAT 11.2/0-4 (Alire) complains

```
Error: invalid instruction shape
-- `vrintn.f64 s14,s14'
```

presumably because the operands are S registers rather than double precisions D registers. Is this a bug or have we misunderstood something?

From: Ahlan Marriott <ahlan@marriott.org>
Date: Fri, 24 Nov 2023 01:09:38 -0800

The solution is to use %P to access the parameters constrained using “w”. Try as I might I can’t find this wonderful secret documented anywhere.

I stumbled on the solution in the NXP forum where jingpan replied to a question on how to use the ARM VSQRT instruction for double. When using the inline assembler from C and using named parameters you need to access parameters constrained by “w”, i.e. D registers using %P[name] rather than %[name] as everywhere else.

Using positional parameters one needs to use %Pn rather than %n

And yes it must be a capital P.

I fail to understand why one needs to do this because surely the assembler already knows that the parameter has been constrained to a D register - but I guess this is just an additional quirk to an already very quirky assembler.

My GNAT Ada code to implement the Unbiased_Rounding attribute efficiently using the VFLOATN instruction is therefore

```
subtype T is Long_Float;
function Unbiased_Rounding (X : T) return
T is
   Y : T;
begin
   Asm (“vrintn.f64 %P0,%P1”,
   Outputs => T’asm_output (“=w”, Y),
   Inputs => T’asm_input (“w”, X));
return Y;
end Unbiased_Rounding;
```

Of course we wouldn’t have to resort to assembler at all had there been a built-in intrinsic for VFLOATN as there is for all the other VFLOAT instructions. But I guess that is hoping for too much.

GNAT.Source_Info Volatile and SPARK

From: Kevin Chadwick <kc-usenet@chadwicks.me.uk>
Subject: GNAT.Source_Info Volatile and SPARK
Date: Fri, 8 Dec 2023 18:28:24 -0000
Newsgroups: comp.lang.ada

I guess the SPARK annotations in GNAT.Source_Info mark them as Volatile_Functions for good reason.

I'm not sure how to handle using them in SPARK. They produce compile time known constants but I guess SPARK does not know e.g. the String length.

I use them in a logging function which is everywhere. So I get error “Volatile function call as actual is not allowed in SPARK” when calling GNAT.Source_Info.Source_Location as a logger’s parameter. Perhaps I should just avoid using them for SPARK compatibility? I can get by with GNAT.Source_Info.Line which only produces warnings and not the above error but it is not ideal.

I can use the function File as a package global constant. Any other ideas?

From: Kevin Chadwick <kc-usenet@chadwicks.me.uk>
Date: Sat, 9 Dec 2023 14:16:17 -0000

> I can use the function File as a package global constant. Any other ideas?

I shall go with doing the above per package for Gnat.Source_Info.File and wrapping the Gnat.Source_Info.Line procedure with one marked with Global => null.

Where would I suggest that Global => null be added to Line?

From: Kevin Chadwick <kc-usenet@chadwicks.me.uk>
Date: Sat, 9 Dec 2023 14:33:22 -0000

Doh… Of course I can't wrap Line, ha ha. If I want the right line.

From: Jeffrey R.Carter <spam.jrcarter.not@spam.acm.org.not>
Date: Sat, 9 Dec 2023 15:57:41 +0100

> Doh… Of course I can't wrap Line

Perhaps

```
private with Gnat.Source_Info;
package Source_Line_Info with
SPARK_Mode is
   function Line ... with Global => null;
private -- Source_Line_Info
   pragma SPARK_Mode (Off);
   function Line ... renames
    Gnat.Source_Info.line;
end Source_Line_Info;
```

(Untested)

From: Kevin Chadwick <kc-usenet@chadwicks.me.uk>
Date: Sat, 9 Dec 2023 15:13:28 -0000

> (Untested)

Interesting, Thanks. I might just use random identifiers. With the added benefit of knowing it will work with any runtime, any platform and any compilation options.

Beaujolais Challenge

From: jklsemicolon@f172.n1.z21.fsxnet (Jklsemicolon)
Subject: Beaujolais Challenge
Date: Sun, 10 Dec 2023 12:34:03 +1300
Newsgroups: comp.lang.ada

More than twenty years ago as a high schooler digging into the stacks at a community college library, I came across a book on Ada where a chapter epigraph referenced a bug bounty where the finder of some variety of bug in the Ada language specification would receive a case of wine. Does this ring any bells? I realize that this is quite vague, but I didn't have the CS background then to appreciate what I was reading, and events have taken me quite far from that shelf on that day.

... We all live in a yellow subroutine...

From: J-P. Rosen <rosen@adalog.fr>
Date: Sun, 10 Dec 2023 19:39:54 +0100

Sure. Ichbiah bet that the addition or removal of a use clause could cause compilation errors, but could not give a working program with a different meaning (a different resolution).

John Goodenough came up with such a case (a very contrived case, involving several levels of generics). I'm not sure that Ichbiah offered the bottle... The so-called beaujolais effect was fixed in Ada95.

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Date: Sun, 10 Dec 2023 19:27:26 -0000

>The so-called beaujolais effect was fixed in Ada95.

See also: https://en.wikipedia.org/wiki/Beaujolais_effect.

And the reference given on that wiki-page: https://web.archive.org/web/20060823054957/http://www.adaic.com/learn/oldfaqs.html#beaujolais

From: Randy Brukardt <randy@rrsoftware.com>
Date: Tue, 12 Dec 2023 03:23:41 -0600

>...The so-called beaujolais effect was fixed in Ada95.

It's still something that is talked about today when new Ada features are proposed; we don't want to reintroduce it, or the related "Ripple effect" (which is associated with "with" clauses, and is named for a cheap American wine brand circa 1980).

Map Iteration and Modification

From: Drpi <314@drpi.fr>
Subject: Map iteration and modification
Date: Thu, 28 Dec 2023 14:53:16 +0100
Newsgroups: comp.lang.ada

I need to delete nodes from a Hashed_Map. I don't know which nodes to delete in advance. I have to iterate on the Map keys and delete the nodes which fulfill a condition. From the LRM I understand I can't delete nodes within a loop iterating the Map nodes. That makes sense. What's the recommended way of doing this? Iterate the Map and temporarily store the key nodes to be deleted then delete the nodes from the key list?

From: Drpi <314@drpi.fr>
Date: Thu, 28 Dec 2023 14:59:07 +0100

> Iterate the Map and temporarily store the key nodes to be deleted then delete the nodes from the key list?

Not clear. Rephrasing it.

Using 2 steps by iterating the Map and temporarily store the keys of nodes to be deleted then delete the Map nodes using the key list?

From: Randy Brukardt <randy@rrsoftware.com>
Date: Thu, 28 Dec 2023 21:08:47 -0600

If the keys are messy to save (as say with type String), it might be easier to save the cursor(s) of the nodes to delete. You would probably want to use a cursor iterator (that is, "in") to get the cursors. Code would be something like (declarations of the Map and List not shown, nor is the function Need_to_Delete which is obviously application specific, Save_List is a list of cursors for My_Map, everything else is standard, not checked for syntax errors):

```
Save_List.Empty; -- Clear list of saved
                 -- cursors.
-- Find the nodes of My_Map
-- that we don't need.
for C in My_Map.Iterate loop
   if Need_to_Delete (My_Map.Element(C))
   then
      Save_List.Append (C);
      -- else no need to do anything.
   end if;
end loop;
-- Delete the cursors of the nodes we don't
-- want anymore.
for C of Save_List loop
   My_Map.Delete(C);
end loop;
```

From: Drpi <314@drpi.fr>
Date: Fri, 29 Dec 2023 14:53:53 +0100

That's what I did but I saved the keys (String) instead of the cursors.

Does it make a difference? Performance maybe?

From: Randy Brukardt <randy@rrsoftware.com>
Date: Sat, 30 Dec 2023 00:29:07 -0600

> That's what I did but I saved the keys (String) instead of the cursors. Does it make a difference? Performance maybe?

It certainly will make a performance difference; whether that difference is significant of course depends on the implementation. There's two parts to it (one of which I thought of yesterday and the other which I forgot):

(1) The cost of storing keys vs. storing cursors. Cursors are going to be implemented as small record types (canonically, they are two pointers, one to the enclosing container and one to the specific node/element). A key can be most anything, and storing that can be more costly.

(2) The cost of looking up a key. A map is a set of nodes, and there needs to be some operation to associate a key with the correct node. Those operations take some time, of course: for a hashed map, the key has to be hashed and then some sort of lookup performed. Whereas a cursor generally contains an indication of the node, so the access is more direct.

For a lot of applications, this difference won't matter enough to be significant. But I'd probably lean toward using cursors for this sort of job as that would minimize performance problems down the line. (Of course, if the container gets modified after you save the cursors, then they could become dangling, which is a problem of it's own. If that's a possibility, saving the keys is better.).