Minutes of Electronic ARG Meeting 62P

5 May 2022

Attendees: Raphael Amaird, Steve Baird, John Barnes (left at 12:35), Randy Brukardt, Jeff Cousins (arrived at 10:36), Gary Dismukes, Claire Dross, Bob Duff, Brad Moore, Jean-Pierre Rosen, Ed Schonberg, Justin Squirek, Tucker Taft, Tullio Vardanega.

Observers: None.

Meeting Summary

The meeting convened on Thursday, 5 May 2022 at 10:32 hours EDT and adjourned at 13:31 hours EDT. The meeting was held using Zoom. The meeting covered much of the agenda.

AI Summary

The following AIs were approved:

AI22-0030-1/02 Array iterators of slices (13-0-0)
AI22-0035-1/02 Add “not null” to some Ada.Strings subprograms (13-0-0)

The following AIs were approved with editorial changes:

AI12-0438-1/01 Rewording to remove “must” (14-0-0)
AI12-0439-1/01 Rewordings to replace "might" or "could" with "can" or "may" (14-0-0)
AI12-0440-1/01 Rewording of notes to replace other words with “can” (14-0-0)
AI12-0441-1/02 Introductory wording changes (14-0-0)
AI12-0442-1/01 Rewordings of notes (14-0-0)
AI12-0443-1/01 Convert glossary into a separate Terms and Definitions clause (14-0-0)
AI12-0444-1/02 Rewordings to remove “need not” (14-0-0)
AI12-0445-1/02 Rewordings of normative material (14-0-0)
AI22-0028-1/03 Program_Error for unchecked union equality (13-0-1)
AI22-0039-1/02 Clarify 8.3(26/2) (13-0-0)

The intention of the following AIs were approved but they require a rewrite:

AI22-0019-1/01 Freezing of aspect specifications (13-0-0)
AI22-0029-1/02 External_Tag collisions (14-0-0)
AI22-0038-1/01 Introduction to 7.3.2 (13-0-0)

The following AIs were discussed and voted No Action:

AI22-0035-2/01 Add “not null” to type Character_Mapping_Function (13-0-0)

Detailed Minutes

Welcome

Steve welcomes everyone.

Apologies

Arnaud Charlet has a family conflict. Richard Wai was bringing home his wife and brand-new daughter (born Tuesday). Jeff Cousins has a chest infection and has lost his voice (he joined anyway and participated primarily via the chat).

Previous Meeting Minutes

There were no comments on the minutes: Approve minutes: 14-0-0.

Date and Venue of the Next Meeting

Randy proposes Thursday, June 23, 2022 for our next meeting. There being no objections, we adopt that date.

AI Format Modernization

We're just starting to use the new formats and procedures set up by the User Community Input process. Because the FDIS has a rapidly approaching deadline, the editor has not had time to update the tools and convert all of the existing AI22s to the new formats. We also still have to set up AIs at Github.

The basic approach will be that AIs are created and edited on the Google Drive at arg.adaic.org. Snapshots will be taken for version control, indexing, documents, and long-term access purposes at least before each meeting (and possibly more often when that makes sense). There will be direct HTML versions available on Ada-Auth.org (from the snapshots).

We're just starting this process now, and the snapshots are still being processed by hand, most of the tools and scripts have yet to be updated or created. So it's currently a mess. Each AI for this meeting has an indication in the agenda where it can be found.

SPARK Language Study Group

Steve and Tucker ask whether the ARG site should host a SPARK Language Study Group. AdaCore would like to use the ARG process and sites to evolve SPARK. Tucker notes that SPARK and Ada are somewhat interrelated and features have and could migrate in both directions. So it seems natural to let them use the ARG process and sites.

No one voices an objection to this.

Working Group Report – UCI (User Community Input)

Since Richard could not make the meeting, we skip his report.

FDIS

As noted in the agenda, we have received a number of comments from ISO about the state of the Ada Standard. These will need to be corrected in the FDIS document, due by May 30th. The largest is the use of language, some of which is not compliant with the Directives, Part 2.

Randy notes that we don't have to make any language changes to the Ada Reference Manual. (We previously decided that we would not renumber the clauses, keeping the current organization of the Ada Reference Manual.) But it would be easier for future maintenance to keep the Ada Reference Manual and ISO Standard as close as possible. So he believes we should make changes to the Ada Reference Manual as well as the ISO Standard as much as possible. Tucker concurs. No one voices any objections to this plan.

Someone asks if we have a summary of the rules for future use. Randy notes that there is such a summary in the discussion of several of the new AIs. It is suggested that these rules be integrated into the (new) AI style guide. Randy will take an action item to do that.

Parting Shot

At the end of the meeting, Ed Schonberg asks if the meeting will be dated "Can 5th", since ISO will not let us use "May". After a bit of confusion, everyone laughs.

The editor notes that one of the hits for "may" in non-normative text was an enumeration type example using months of the year. He almost changed that to "Can".

Unfinished Action Items

There are a few unfinished action items (Tucker Taft: OpenMP Technical Report; AI22-0034-1; Edward Fish: AI22-0022-1). We did not discuss these. Additionally, three people still owe ACATS Tests: Ed Schonberg, Justin Squirek, and Tucker Taft.

Current Action Items

The combined unfinished old action items and new action items from the meeting are shown below.

Randy Brukardt:

- Add the wording rules from the JTC1 Directives, Part 2, to our newly published style guide.

Editorial changes only:

- AI12-0438-1
- AI12-0439-1
- AI12-0440-1
- AI12-0441-1
- AI12-0442-1
- AI12-0443-1
- AI12-0444-1
- AI12-0445-1
- AI22-0028-1
- AI22-0039-1

Edward Fish:

- AI22-0022-1

Ed Schonberg:

- ACATS C-Test(s) for filters

Justin Squirek:

- ACATS C-Test(s) for Wide versions of Command_Line, etc.

Tucker Taft:

- AI12-0346-1 – also, construct the Technical Report suggested by this AI.
- AI22-0019-1
- AI22-0029-1
- AI22-0034-1
- AI22-0038-1
- ACATS C-Test(s) for container aggregates

Detailed Review

The minutes cover detailed review of Ada 2012 AIs (AI12s) and Ada 2022 AIs (AI22s). The AI12s and AI22s are presented in numeric order, which is not necessarily the order in which they were discussed. Votes are recorded as "for"-"against"-"abstentions". For instance, a vote of 6-1-2 would have had six votes for, one vote against, and two abstentions.

If a paragraph number is identified as coming from the working Ada 202x AARM, the number refers to the text in draft 32 of the Ada 202x AARM (the submission draft). Paragraph numbers in other drafts may vary. Other paragraph numbers come from the final consolidated Ada 2012 AARM; again the paragraph numbers in the many drafts may vary.

Detailed Review of Ada 2012 AIs

AI12-0438-1/01 Rewording to remove “must”

Introduction 6/2: “{are}[need] not [be]”. And change “ought to” to “is” (“is visible”).

Typos will or have been e-mailed.

Approve AI with changes: 14-0-0.

AI12-0439-1/01 Rewordings to replace "might" or "could" with "can" or "may"

We decide not to look at typos during the meeting. Gary and Brad will send typos, Jeff already has.

No one has any substantive comments on this AI.

Approve AI with changes: 14-0-0.

AI12-0440-1/01 Rewording of notes to replace other words with “can”

3.9.4(27/2) is missing “may” to “can”.

Typos will or have been e-mailed.

Approve AI with changes: 14-0-0.

AI12-0441-1/02 Introductory wording changes

Gary notes that comment #3 does not match the directives. Randy checks and that is the exact wording as found in the comments file. We should add a note somewhere in the AI that the comment seems incorrect.

Typos will or have been e-mailed.

Approve AI with changes: 14-0-0.

AI12-0442-1/01 Rewordings of notes

Brad does not like “is allowed to” in 5.5.3(21/5) and 5.5.3(27/5). He thinks it is too weak. This is official wording for a permission, and ISO wants to stamp out nuance as they claim it confuses non-native English speakers. Randy notes these are normative wording paragraphs and thus are in the wrong AI, they should be in AI12-0445-1.

Typos will or have been e-mailed.

Approve AI with changes: 14-0-0.

AI12-0443-1/01 Convert glossary into a separate Terms and Definitions clause

Tucker notes that “default initial condition” is written wrong. The “a default initial condition is” part should be deleted.

Gary does not like the wording of “record extension”. “...extends another type [by adding]{optionally with} additional components”

Typos will or have been e-mailed.

Approve AI with changes: 14-0-0.

AI12-0444-1/02 Rewordings to remove “need not”

Gary: In 13.3(30/2) both “which” should be “that”. Same in 13.3(30.1/2).

Typos will or have been e-mailed.

Approve AI with changes: 14-0-0.

AI12-0445-1/02 Rewordings of normative material

Move those two paragraphs from AI12-0442-1.

Surprisingly, no one had any typos for this AI.

Approve AI with changes: 14-0-0.

Detailed Review of Ada 2022 AIs

AI22-0019-1/01 Freezing of aspect specifications

Randy notes that we have some relatively complicated freezing rules for aspect specifications, and we did not want to duplicate those.

There are some font errors in the wording.

Ed notes that “evaluated” is spelled wrong in the discussion.

Jeff is confused by “causes freezing upon the freezing of”. Various rewording are proposed, but none really work well. Tucker will try to reword. We’re out of time for today’s meeting, so he’ll do that off-line.

Approve intent of AI: 13-0-0.

AI22-0028-1/03 Program_Error for unchecked union equality

Tucker’s last name is misspelled in the !appendix.

Ed note that in the !summary, “checked” should be “evaluated” or maybe “applied”.

Tucker rewrites the summary in the Google Doc. Some wordsmithing ensues, but eventually everyone is happy.

Approve AI with changes: 13-0-1. John abstains.

AI22-0029-1/02 External_Tag collisions

Jean-Pierre Rosen suggests using a list (instead of “if not”):

> Program_Error is raised at one of the following places:
> - an attribute_definition_clause for S’External_Tag;
> - an attribute_definition_clause for T’External_Tag;
> - the freezing point of S;
> - the freezing point of T.

Is “declared” is the right word in paragraph 75.1? Steve wonders if a type that ceases to exist (because the subprogram was left) still can cause this failure.

Randy notes that we have intended to allow two different implementations here: a fully static implementation with bind-time construction of an external tag table, and a runtime scheme where the elaboration of a type registers the external tag with a manager in the runtime. We want this rule to be compatible with both implementations.

Tucker suggests qualifying the (new) start:

> Program_Error is raised if S and T exist simultaneously at one of the following places:

But that doesn’t work for the fully static (bind-time) implementation.

> If both S and T exist concurrently in a given partition, Program_Error is raised at one of the following places:

Another try:

> At a point when both S and T exist in a given partition and S’External_Tag is the same as T’External_Tag, Program_Error is raised at one of the following places:

Steve notes that we now have two points that we’re talking about which is weird.

> If both S and T exist in a given partition and S’External_Tag is the same as T’External_Tag, Program_Error is raised at one or more of the following places:

And then add an Implementation Permission to raise Program_Error if both are part of the same partition statically. Obviously, this latter part is not a formal proposal.

Tucker will take this AI.

Approve Intent: 14-0-0.

AI22-0030-1/02 Array iterators of slices

Approve AI: 13-0-0.

AI22-0035-1/02 Add “not null” to some Ada.Strings subprograms

Tucker also has an alternative (AI22-0035-2), where he changes the type rather than all of the uses.

Claire notes that adding “not null” to the type would cause any existing objects declared of the type but that are not initialized to raise Constraint_Error. That’s a pretty bad incompatibility (the RM would call it an “inconsistency”, since it occurs only at runtime). Tucker notes that a common use case would be to declare an object, and then initialize to a mapping function with some complex code, and then use it. Such a use fails with the change suggested in the alterative. So we abandon the alternative.

Approve AI: 13-0-0.

AI22-0035-2/01 Add “not null” to type Character_Mapping_Function

No action: 13-0-0.

AI22-0038-1/01 Introduction to 7.3.2

Tucker thinks this got a little wordy. Steve suggests dropping the “on any objects”.

> “... but is not enforced [on any objects] while inside an operation of the package...”

The last sentence should be a note; Steve says it is isn’t true when there are globals. Jeff notes that it isn’t true for default initialization.

Randy notes that you get enforcement inside the package when an object is default initialized. Steve notes that it also gets checks when you call a visible operation inside of the package. So drop the entire part about “but not enforced”.

The wording “every object” should be “objects”, since there are unlikely cases not checked.

Is ‘every type” OK? “any type covered by the class-wide type”.

There is too much change here, Tucker will take this back.

Approve Intent: 13-0-0.

AI22-0039-1/02 Clarify 8.3(26/2)

Gary notes that there should not be a hyphen in “non-overridable”. It’s odd that there is a hyphen in the second use but not the first identical rule.

Approve AI with change: 13-0-0.