

Minutes of ARG Meeting 63F

31 October 2024

Attendees:

Raphael Amiard (10:40-11:00), Steve Baird, John Barnes, Randy Brukardt, Jeff Cousins (leaves 13:04), Gary Dismukes (leaves 12:10), Edward Fish, Christoph Grein, Niklas Holsti, Brad Moore, Alejandro Mosteo, Jean-Pierre Rosen, Justin Squirek, Tucker Taft, Tullio Vardanega, Richard Wai.

Observers:

None.

Meeting Summary

The meeting convened on Thursday, 31 October 2024 at 10:36 hours EDT and adjourned at 13:30 hours EDT. The meeting was held using Zoom. The meeting covered most of the new and updated AIs on the agenda.

AI Summary

The following AI was approved:

AI22-0120-1/01 Relax array aggregate restriction on nonstatic choices with others (10-1-2)

The following AIs were approved with editorial changes:

AI22-0113-1/03 Usage advice (part 3) (13-0-2)

AI22-0114-1/01 A “main program” is not a defined term (15-0-0)

AI22-0115-1/03 Usage of the prefix of the Valid attributes (15-0-0)

AI22-0116-1/02 Discard_Names and Put_Image (15-0-0)

AI22-0118-1/02 Put_Image for derived types with new discriminants (12-0-2)

AI22-0119-1/02 Canonical order should be defined once (13-0-1)

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

AI22-0101-1/04 Valid_Scalars attribute is defined (13-0-0)

AI22-0121-1/01 Internal calls to contracts of protected types (15-0-0)

Detailed Minutes

Welcome

Steve welcomes everyone to this meeting.

Apologies

Bob Duff reported that he was unable to attend.

Previous Meeting Minutes

There were no comments on the minutes: Approve minutes: 15-0-0.

Date and Venue of the Next Meeting

Our next electronic meeting is proposed for Thursday, January 9 with the usual time (10:30-13:30 EST [-5 UTC]) and method (Zoom). No objections are noted. So the proposed date is scheduled.

User Community Input Working Group Report

Nothing to report for the UCI. Tucker notes that there is a now high-performance link to the RM. More documents will be made available there soon.

Future ARG Funding

Tullio reports that he is working on a report for the December 2024 WG 9 meeting, and he will have more to say then. Ed asks how much money is needed. Tullio says that we need three roles (WG 9 convenor, ARG chair, ARG editor). And we need to figure out how to handle those roles. Tucker notes that the WG 9 convenor role is primarily travel costs. Randy is asked what the cost is. Randy says that he spends about 500 hours per year plus internet and the INCITS fee (\$1500). Volunteers could probably do some of the work, but there is value to having someone whose job is keeping everything organized.

AdaCore proposed features

Raphael talks about AdaCore's process for enhancements. They start with an RFC, these usually don't include any RM-style wording. They're more of an "informal formal" description. Then it can be prototyped. Often there is feedback from the prototype to improve/change the description. Once the prototyping is finished, it then will get added to the GNAT documentation. This is not a fixed group of people, but rather the people who are most knowledgeable about a particular area.

Tucker notes that we can use some of these enhancements as the basis for RM updates. They have already been prototyped, so they should be relatively easy to finish through our processes. He mentions that it would be helpful for the most involved people to come to the appropriate ARG meeting to provide information about the changes. Raphael thinks that should be possible.

Raphael notes that AdaCore will welcome ARG feedback on these features, including potential changes. They recognize that not every interaction is understood by their team.

Upcoming Corrigendum

Randy notes that the plan is to have a draft of the Corrigendum document for review well before the next ARG meeting. Tucker asks whether the Corrigendum is a list of changes. Randy says that the drafting rules have little to say about the form of the body of the Corrigendum, so we've always used the list of changes format. Of course, we'll have an updated RM to use for most purposes.

Randy intends to write a short justification for every included change. This is intended to make it easier to respond to any complaints from the ISO editors about inclusions.

Tucker asks what will happen to the changes not included. Those will be gathered in the form of an Amendment to form the start of future work. Randy intends to keep those in the draft RM, just as /7 changes (post Corrigendum). That's easy to do in the source, just a change to the version number of the @Chg commands is needed.

Unfinished Action Items

We did not discuss any of the unfinished action items.

Current Action Items

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

Steve Baird:

- AI22-0076-1 (with help from Tucker Taft)
- Open a Github issue on the term normal (see discussion of AI22-0115-1)

Randy Brukardt:

- AI22-0121-1

Editorial changes only:

- AI22-0113-1
- AI22-0114-1
- AI22-0115-1
- AI22-0116-1
- AI22-0118-1
- AI22-0119-1

Jeff Cousins:

- Assist Ed with AI22-0022-1

Edward Fish:

- AI22-0022-1

Justin Squirek:

- ACATS C-Test(s) for filters

Tucker Taft:

- AI22-0063-1 (Split work with Richard Wai)
- AI22-0071-2
- AI22-0076-1 (assist Steve Baird)

Richard Wai:

- AI22-0063-1 (Split work with Tucker Taft)

Detailed Review

The minutes cover detailed review of Ada 2022 AIs (AI22s). The 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 202y AARM, the number refers to the text in draft 1 of the Ada 202y AARM. Paragraph numbers in other drafts may vary. Other paragraph numbers come from the final Ada 2022 AARM; again the paragraph numbers in the many drafts may vary.

Detailed Review of Ada 2022 AIs

AI22-0101-1/04 Valid_Scalars attribute is defined

We discussed this last time. Steve had some questions, and those have been answered, and the wording was updated and the questions were removed.

The erroneous execution section should be a note or perhaps a Usage.

Randy thinks that the prefix being abnormal is different than for ‘Valid, because we cannot “renormal” a composite object as we do for ‘Valid. So we cannot use the same rules that we use for ‘Valid. Steve wonders if all bad objects are abnormal. We’ll need to check that.

Randy will try to figure this out and update the AI accordingly.

Approve intent: 13-0-0.

AI22-0113-1/03 Usage advice (part 3)

Christoph explains the AI.

Tucker suggests that it is OK to have advice that there exists an alternative. We don't need to say that it is preferable. Alternative wording is explored. Steve suggests "sometimes preferable" and everyone laughs.

We decide to leave the wording alone, and just add to the discussion to explain that we decided that it was better as it was then any of the alternatives that we explored.

The other changes are fine.

Approve AI with changes: 13-0-2 (John and Brad abstained)

AI22-0114-1/01 A "main program" is not a defined term

Randy notes this came out of an editorial comment from John. He found several additional uses beyond the one originally noted by John.

Tucker asks if the AARM note is new. Randy confirms that it is. He had to chose between replacing "main program" with either "partition" or "main subprogram" in the case of temporary files, and the AARM note explains the choice made. Since "main program" is undefined, the exact rule intended is unknown. Tucker adds the missing insertion brackets.

This change could be inconsistent with some implementation, but it is more likely that implementations work as defined here because of the overhead needed to determine the "last" partition terminated for the "main subprogram" interpretation.

Approve AI with changes: 15-0-0.

AI22-0115-1/03 Usage of the prefix of the Valid attributes

Randy explains that these rules were only stated in notes, but they cannot be derived from any other rules. Indeed, given that we need a "notwithstanding" rule to get them, they are nonsense as notes.

The remaining note text should still say "Note 2". This is the consequences of the non-evaluation rule. We discuss if this should be Usage advice, but the discussion ends without a conclusion.

Steve thinks "normal" isn't testable. It happens when some assignment is aborted, and we aren't going to track that with a bit. We start to remove that part, but then Randy notices that X'Valid says that it is True only if it is normal. That sounds weird.

Steve thinks it still should be erroneous to read abnormal objects. Tucker disagreed, this attribute only works on scalars, and it effectively makes objects normal if it returns True (even if abnormal before). The AARM notes and examples make it pretty clear that was the intent.

Tullio wonders if "normal" is really "not abnormal". The term "normal" is not defined, and it is not obvious that it is the inverse of "abnormal" in this context. So this all really should say "not abnormal". But "normal" is used widely in this clause, so that could be a lot of change and some problems with double negatives.

We suggest opening a Github issue on this point. Steve will take action item to do this. [Editor's note: The word "normal" is italicized as a term in 13.9.1, but it is indexed as "normal state of an object". Just defining "normal" by itself is problematic, as it occurs in a number of other terms, such as "normal termination" and "normal completion". The term probably should be "normal state" (italicizing "state" in 13.9.1), and then possibly using in "normal state" rather than "normal" appropriately in the wording.]

Approve AI with changes: 15-0-0.

AI22-0116-1/02 Discard_Names and Put_Image

The AARM note is incorrect, it should say no effect on the Wide_Wide_Image attribute (not no effect on anything). We'll trust the editor to fix this non-normative text.

Approve AI with changes: 15-0-0.

AI22-0118-1/02 Put_Image for derived types with new discriminants

Tucker wrote this one. He explains that if the discriminants change, the default Put_Image rule doesn't make sense.

So he redid the rules so that if there isn't an ancestor with a specified Put_Image, then the usual component-by-component record type is displayed.

Jeff asks if this is an incompatibility. Randy says it certainly is, it should be mentioned in the !discussion. Tucker adds that to the Google Doc version of the AI.

Approve AI with changes: 12-0-2. Jeff, Brad abstain.

After the vote, Ed wonders if there would be value to a T'Aggregate which gives the underlying representation of an object. It would be essentially defined like the default version of Put_Image, but no overriding is possible. Ed thinks this would be useful for debugging, for hand-written version of composite Value routines, and for meta programming proposed. The latter is out of bounds here now, and otherwise we're not sure of the value. It should be proposed separately.

AI22-0119-1/02 Canonical order should be defined once

Tucker explains one choice was to remove "canonical order" from Put_Image. The other is to expand the definition. He tried the latter here.

Steve wonders about the order for type extensions. That is defined in earlier paragraphs of 3.4.

Randy notes that the new visibility wording in 3.4 is redundant. That should be properly defined in 8.2 or 8.3 (if it's not, that's a different problem). Tucker makes this change in the Google Docs version of the AI.

Jean-Pierre wonders about the specific exception for Fortran. Tucker notes that he now allowed that to be specified for other language conventions. He makes a minor wording change to clarify this.

Approve AI with changes: 13-0-1 (Randy abstains)

AI22-0120-1/01 Relax array aggregate restriction on nonstatic choices with others

Tucker explains that this was raised by a user, it seems natural. We don't plan to go further, since having multiple choices would require some sort of runtime non-overlap check. But others choices don't need any such check (they don't overlap by definition).

Steve worries about nonstatic subtype choices with a discontinuous Static_Predicate. Those could complicate the implementation a lot.

Tucker says this is equivalent to:

```
[for I in Applicable_Index_Constraint => (if A then True else False)]
```

[Editor's note: After the meeting, Tucker reported that 3.2.4(28/3) makes it illegal for a dynamic array aggregate choice to have any predicate expressions that apply. So this is not a real issue.]

Randy remains concerned about the implementation cost. Transforming an aggregate as suggested would generate pretty terrible code (optimization would be difficult). The cost is high enough and the value is low enough that the cost/benefit ratio is on the edge of too high.

Approve AI: 10-1-2. Opposed: John; Abstain: Randy, Jean-Pierre.

AI22-0121-1/01 Internal calls to contracts of protected types

Randy explains that we already had a rule that internal calls are not allowed in preconditions. So he just expanded that rule to cover these additional aspects. One might think that the rule needs to cover `Type_Invariant`, but that aspect is only allowed on private types (and there are no private protected types).

Tucker wonders how you could write an external call. Tucker thinks that `T1.Get_X` is actually an internal call. He says that because `T1.Get_X` is an expanded name, and expanded names mean that the call is internal. Steve suggests that you could easily use a wrapper function that is passed the current instance to make an external call.

Randy thinks that `T1` is an object of the current instance, and this is a `selected_component` rather than an expanded name. Various AIs updated 8.6 to that effect, and this was one of the reasons. Tucker is skeptical.

Brad notes that we should have an AARM note if a wrapper is needed (assuming that is true).

We decide that even though the wording is correct in any case, we need to resolve this question about how one can write an external call and update the discussion accordingly. This is the sort of thing we shouldn't try to resolve during a (short) virtual meeting.

Approve intent: 15-0-0.