Minutes of Electronic ARG Meeting 62E

30 July 2020

Attendees: Raphael Amiard , Steve Baird, Randy Brukardt, Arnaud Charlet, Jeff Cousins, Gary Dismukes, Claire Dross, Bob Duff, Brad Moore, Jean-Pierre Rosen, Ed Schonberg, Justin Squirek, Tucker Taft, Tullio Vardanega, Richard Wai.

Observers: None.

Meeting Summary

The meeting convened on Thursday, 30 July 2020 at 10:32 hours EDT and adjourned at 13:30 hours EDT. The meeting was held using Google Meet. The meeting covered almost all of the AIs on the agenda.

AI Summary

The following AIs were approved with editorial changes:

AI12-0374-2/02 Fixes for Nonblocking (9-0-6)
AI12-0380-1/04 Fixups for Global annotations (8-0-7)
AI12-0386-1/01 Still more presentation issues (15-0-0)

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

AI12-0355-1/02 Generalized aspect specifications (11-0-4)
AI12-0378-1/05 View conversions and out parameters of access types revisited (14-0-1)

The following AI was discussed and assigned to an editor:

AI12-0384-1/02 Fixes for Put_Image and Text_Buffers

The following AIs were discussed and placed on hold:

AI12-0257-1/02 Generalize prefix views (15-0-0)
AI12-0349-1/01 Add “parallel” to aggregate iterators (13-0-2)
AI12-0360-1/02 Procedural iterators for generic procedures (15-0-0)

Detailed Minutes

Welcome

Steve welcomes everyone.

Previous Meeting Minutes

A couple of typos were fixed in these minutes. Approve minutes: 15-0-0.

Date and Venue of the Next Meeting

Following our approximately 6 week schedule, Randy had suggested Wednesday, September 9th for our next meeting. Jean Pierre has a training session on 9-10-11 and 16-17-18, and he also has other commitments on the 14th and 15th. Tucker has a doctor appointment on September 8th, and September 7th is Labor Day in the US (a major holiday).

We decide to have our next electronic meeting Wednesday, September 9, 10:30-1:30 EDT as we can’t find a better date.

Minimum time between agenda posting and meeting

Arnaud would like a longer time for reviewing AIs before a meeting. He would like a week (or at a minimum 5 days) for review. Richard, Raphael, and others agree. Tucker suggests that the homework deadline be 24 hours before the agenda deadline.

We vote on this proposal (5 days before the meeting for the agenda, 6 days for homework and other submissions).

Approve proposal: 15-0-0.

Reference Manual Review Plan

Randy says that he didn't quite finish applying all of the changes to the RM, and given that more changes will be approved today, there didn't seem to be any reason to rush. (Additionally, his primary computer is not working which would have added an additional challenge.)

He expects to have a draft RM and assignments in a couple of weeks after this meeting. We discuss the time for review; everyone over-optimistically says that they'll only need two weeks. Randy says that he is planning a four week deadline (probably just before the next meeting), but of course getting reviews done early is always appreciated.

He expects that we will have a substantial number of AIs to resolve issues raised during the review, and those will take time to work through (his guess is three meetings).

Randy asks members to send preferences for review if they have any. He notes that he will try to assign different clauses to members than in the last (2019) review.

Unfinished Action Items

There are five unfinished action items (Steve Baird, AI12-0016-1; Ed Fish: AI12-0215-1; Tucker Taft: OpenMP Technical Report; Tucker Taft: Propose improvements to the definition of aspects). We did not spend any time talking about these.

Current Action Items

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

Steve Baird:

- AI12-0016-1
- AI12-0378-1

Randy Brukardt:

Editorial changes only:

- AI12-0374-2
- AI12-0380-1
- AI12-0386-1

Arnaud Charlet:

- Create and circulate a list of Ada 202x features that AdaCore is not currently planning to implement (see discussion of AI12-0360-1).

Edward Fish:

- AI12-0215-1 (with help from Raphael Amiard)

Tucker Taft:

- AI12-0346-1 – also, construct the Technical Report suggested by this AI.

- AI12-0355-2 (new, simpler alternative).
- AI12-0384-1
- Propose improvements to the definition of aspects (are there two or three kinds of aspects; and which rules apply to each [including subdivisions of each kind]).

Detailed Review

The minutes cover detailed review of Ada 2012 AIs (AI12s). The AI12s 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 25 of the Ada 202x AARM. 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

AI12-0257-1/02 Generalize prefix views

Tucker says that there isn't a detailed proposal. Arnaud would like to "take our time" to get this right.

Jean-Pierre says that is big compatibility issue for syntactic sugar. It doesn't seem worth it. Richard and Randy agree.

Raphael notes that this a very popular feature, so it would be useful to extend it. Tucker notes an Ada-Europe paper uses prefix notation for streaming.

Arnaud notes that AdaCore would like to play with options before pushing forward with a proposal.

Tucker thinks this is too complex to shoehorn in at this point. We want get Ada 202x done this year.

Hold AI: 15-0-0.

AI12-0349-1/01 Add "parallel" to aggregate iterators

Tucker originally thought that a compiler could figure out whether parallelism is needed in cases such as this, since the entire construct is in one place. But since we'd decided that making that an explicit option is important, we ought to support it everywhere.

Richard notes that it would be consistent with the rest of the language to have it allowed for anything using iterator (**for**) syntax. That would also require adding **parallel** to quantified expressions.

Raphael thinks that we would be adding it everywhere is going in the wrong direction. We note that **parallel** can always be ignored (much like inline). Raphael worries that it would keep getting added to constructs.

Straw poll: Do nothing: Arnaud, Claire, Justin, Bob, Steve, Raphael, Ed.
Abstain: Jeff, Gary.

No Action: Tucker, Jeff, Randy abstain. Richard opposes. We never finish this vote when Randy changes his vote to oppose with Richard. He is asked why.

Randy objects to No Action (which means never do): we probably want to do this eventually, maybe not now. It makes sense to determine the usage of **parallel** before continuing, but if it is sufficient, improving consistency certainly should be reconsidered. Richard says that he doesn't want us to forget this change.

Hold AI: 13-0-2. Claire, Arnaud abstain.

AI12-0355-1/02 Generalized aspect specifications

Tucker notes that OpenMP needs tuning parameters for particular constructs.

He describes a feature for dealing with this.

Jean-Pierre asks how the OpenMP implementation would be able to access these specification. Tucker says that's because the implementation of the loop would pass these things. Jean-Pierre notes that the compiler still would need to know that these things are necessary and a mapping; it's not totally in the user's hands.

Randy wonders why this is better than simply defining these as implementation-defined.

Tucker says that one of the goals was to define an API in one's compiler, and then customers could connect/create to bindings to particular lightweight schedulers.

Arnaud thinks it is too late for this iteration.

Randy suggests that we just allow aspect_specification after **parallel**. Tucker says he was just going to make the same suggestion.

That would involve remove everything having to do with generalized_aspect_specifications (all of the wording after 5.5.3(25/5)).

That's too much change to approve today. Make this as a -2 version to preserve this original proposal.

Steve notes that an implementer that doesn't define any aspects for **parallel** would not have to do anything. (There'd be no legal use of an aspect specification, so it wouldn't need to be allowed.)

Approve intent: 11-0-4. Justin, Claire, Arnaud, Raphael abstains.

AI12-0360-1/02 Procedural iterators for generic procedures

Tucker explains the proposal.

Steve thinks it is weird to directly call a generic unit. Tucker does agree that this introduces a case of implicit instantiation, but it is very limited to a particular piece of source.

Steve wonders if rules like accessibility are sufficiently defined. Yes, this is defined by equivalence so those "minor" rules would be defined based on that equivalence.

Raphael mentions that the procedural iterator is not implemented in GNAT. And this is very late. Arnaud concurs.

Gary also says it too late.

Hold AI: 15-0-0.

Jeff requests a list from AdaCore of what Ada 202x features they are not currently planning to implement. Arnaud says that he will do that and he is given an action item to do so.

AI12-0374-2/02 Fixes for Nonblocking

We use the version of the AI from Tucker's Monday e-mail and not the posted version.

Richard complains that the first paragraph of the Legality Rules is unclear about how nesting is handled (see e-mail thread). Tucker says that we'll fix that. Tucker reads the wording change.

Brad wonders if we need to mention invariants in the last paragraph of the !problem. No, those are covered by "assertion expression", and is described in the previous paragraph.

Arnaud notes that AdaCore will not implement it yet, thinks it is too complex and immature.

Approve AI with change: 9-0-6. Ed, Arnaud, Justin, Gary, Bob, Raphael abstain.

AI12-0378-1/05 View conversions and out parameters of access types revisited

Tucker discusses the proposed rules. Unlike the previous proposal, the decision about whether to pass in **null** is made at compile time. If there is a tag check or accessibility check that could possibly fail, then pass in **null**. Otherwise, pass in the original value.

Arnaud is concerned that this rule is very complex. Tucker and Randy note that we have Ada 83/95 compatibility to worry about.

Claire wonders if we should simply abandon compatibility for Ada 202x in this corner case. Passing in **null** always would be better.

Arnaud suggests that we just leave the RM alone. GNAT does not follow the RM in this case. The other compilers run into issues when following the RM. GNAT avoids these problems by passing in **null**, we need to allow that somehow.

Tucker notes that no compiler exactly implements the rule in the manual.

Raphael would prefer to pass in **null** always.

We enumerate possible rules:

(1) Always pass in **null**; (A) for any **out** parameter; (B) for any conversion (other than the implicit one);
(2) Always pass in the original;
(3) Pass null or the actual sometimes.

Richard notes that these sorts of the rules tend to do the right thing; users don't usually worry about the details.

Arnaud asks the original source of this problem. Randy recalls that Steve originated this when thinking about possible issues.

Randy notes that we could declare this a pathology. Arnaud would prefer that.

Steve suggests that we consider an Implementation Permission that implementation can pass **null** in implementation-defined cases if there is an explicit conversion. He would like to give implementers cover for doing something. Tucker notes that implementation-defined requires documentation, so it wouldn't be a surprise to a user.

Steve will take the AI to write a version that does that.

Approve intent: 14-0-1. Bob abstains.

Ed asks if there would be an ACATS test (he hopes not). Randy notes that the ACATS generally does not test implementation choices. Tucker notes that a test that checks that either **null** or the original value is passed in may be valuable. [Editor's note: It's likely that some existing tests would have to be modified this way or removed, since passing **null** would have to be allowed; but it's unlikely that a new test similar to the circulated test program would be created.]

AI12-0380-1/04 Fixups for Global annotations

Tucker describes the reasons for all of these changes.

Steve prefers the longer names of the aspects rather than reserved words **use** and **do**, they're much clearer.

Arnaud notes that Global (in general) is not really ready and he doesn't think it should be in the standard.

Claire agrees generally with Arnaud but does think this is an improvement to Global; it is generally moving in the right direction.

Brad has some typos. He will e-mail them as his connection is iffy and we don't want to spend a lot of time talking about typos anyway.

Approve AI with changes: 8-0-7. Arnaud, Ed, Justin, Jean-Pierre, Gary, Raphael, Brad abstain. (Brad missed part of the discussion and does not think he is qualified to vote.)

AI12-0384-1/02 Fixes for Put_Image and Text_Buffers

Tucker describes the changes.

Bob was concerned about performance. Tucker said he reimplemented it as here and it was not significantly different performance. Bob would like to see the justifications for the changes.

Bob doesn't think that this feature needs to be in the language, AdaCore can provide the functionality to users.

Arnaud notes that this is important to prototype. Tucker notes that no one prototyped the original design (Bob implemented something totally new), so it is hard to compare Bob's results to the original intent.

Claire wonders why this is getting discussed here. This has been going on for a year between Bob and Tucker (and a few others) without a final resolution.

Bob is opposed to approving the AI – he would like to remove the feature – that is user-defined Image. Richard says, speaking as a user, that user-defined image is an important feature.

Arnaud thinks it is too detailed description for this feature. Richard worries that the implementation of Put_Image would not be portable if we left the details to implementers.

Arnaud mentions the detailed description of the default implementation of Image for types. Tucker notes that this AI is just about how someone implements their own Put_Image. This is not at all about the default image.

Tucker will produce a delta and reasoning between Bob's proposal and this one. We agree that we won't vote today.

AI12-0386-1/01 Still more presentation issues

!wording does not match the !corrigendum (and what is in the draft RM). Change the wording to:

```
type Month_Name is (January, February, March, April, May, June, July,
                    August, September, October, November, December);
```

Approve AI with changes: 15-0-0.