Minutes of Electronic ARG Meeting 62D

13 June 2020

Attendees: Steve Baird, Randy Brukardt, Arnaud Charlet, Jeff Cousins, Gary Dismukes, Claire Dross, Bob Duff, Brad Moore, Ed Schonberg, Justin Squirek, Tucker Taft, Tullio Vardanega, Richard Wai (left 1:01 PM EDT).

Observers: None.

Meeting Summary

The meeting convened on Saturday, 13 June 2020 at 10:31 hours EDT and adjourned at 13:31 hours EDT. The meeting was held using Google Meet. The meeting covered many of the AIs on the agenda.

AI Summary

The following AIs were approved with editorial changes:

AI12-0302-1/05 Default Global aspect for language-defined units (10-0-3)
AI12-0344-1/03 Procedural iterator aspects (10-0-2)
AI12-0354-1/01 Semantics of Parallel_Iterators (11-0-2)
AI12-0362-2/01 Attributes for fixed point types (12-0-1)
AI12-0363-1/04 Fixes for Atomic and Volatile (10-0-2)
AI12-0372-1/02 Static accessibility of "master of the call" (12-0-1)
AI12-0377-1/02 View conversions and out parameters of types with Default_Value revisited (12-0-1)
AI12-0379-1/02 More presentation issues (13-0-0)
AI12-0381-1/02 Tag of a delta aggregate (13-0-0)
AI12-0382-1/01 Loosen type-invariant overriding requirements of AI12-0042-1 (13-0-0)
AI12-0383-1/01 Renaming values (12-0-1)
AI12-0385-1/01 Predefined shifts should be static (13-0-0)

The following AI was discussed and assigned to an editor:

AI12-0378-1/03 View conversions of out parameters of access types revisited

The following AI was discussed and placed on hold:

AI12-0362-1/01 Attributes for fixed point types (11-0-2)

Detailed Minutes

Welcome

Steve welcomes us to Santander, Spain. Well, not exactly. (This meeting was in part of the time-slot originally scheduled for the Santander face-to-face meeting, canceled by the COVID-19 pandemic.)

Previous Meeting Minutes

We didn't receive any comments on these minutes. Approve minutes: 13-0-0.

Date and Venue of the Next Meeting

Following our approximately 6 week schedule, Randy had suggested Wednesday, July 29th for our next meeting. Tucker has a conflict on July 29th, he suggests July 28th. Jeff has a conflict on July 28th. So we decide to have our next electronic meeting Thursday, July 30, 10:30-1:30 EDT.

WG 9 report

Steve gives a brief report on the WG 9 meeting that was held the day before this meeting. WG 9 had a resolution about Ada 202x which essentially confirmed our current path. We were asked to carry on taking and resolving feedback. And hopefully be done by the end of the year.

Reference Manual Review Plan

We did a full RM review in January and February 2019, which seems a lifetime ago. We should do another full review shortly, following a similar plan. Randy 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.

Randy hopes to have a new draft of the RM (with PDFs this time) ready for review in about two weeks. He'll then assign roughly equivalent RM text to each ARG member for review, with a 4-6 week deadline. (Even if all of the reviews are completed before the next meeting, it's unlikely that Randy could get them all in shape for the agenda in time for the meeting. So this review is really intended for the September meeting.)

Randy asks members 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 six 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; Tucker Taft and Richard Wai: Review model of Nonblocking). 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

Randy Brukardt:

Editorial changes only:

- AI12-0302-1
- AI12-0344-1
- AI12-0354-1
- AI12-0362-2
- AI12-0363-1
- AI12-0372-1
- AI12-0377-1
- AI12-0379-1
- AI12-0381-1
- AI12-0382-1
- AI12-0383-1
- AI12-0385-1

Edward Fish:

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

Tucker Taft:

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

- AI12-0378-1
- Review the Nonblocking model, including the fixes of AI12-0374-1.
- 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]).

Richard Wai:

- Review the Nonblocking model, including the fixes of AI12-0374-1.

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

AI12-0302-1/05 Default Global aspect for language-defined units

The default is actually "unspecified", not **in out all**.

Tucker notes that we don't want "unspecified" for the Ada library.

Richard notes a missing quote in the first paragraph of !wording.

Steve asks if these lists of packages belong in the Standard. Randy suggests replacing "The Pure Group" with something like "The following units are Pure and do not need a Global specification". And something similar for "The Impure Group". Tucker says that is said at the start, but that is a long way away from the lists.

Claire wonders if the specification of Current_Input/Current_Output as touching everything is too broad. Tucker notes that we don't have a way to limit the global specifications to just objects associated with files. He also notes that Current_Input/Current_Output are not thread-safe, while regular file use is thread-safe. So the connotation is right.

Second paragraph of !wording. Second comma becomes a period. "Alternatively, subprograms may override with..."

Approve AI with changes: 10-0-3. (Gary, Ed, Arnaud abstain)

AI12-0344-1/03 Procedural iterator aspects

Tucker explains the proposals.

Tucker typo: "conf{l}ict" in 2nd para of !discussion.

Approve AI with changes: 10-0-2. Arnaud, Gary abstain.

AI12-0354-1/01 Semantics of Parallel_Iterators

Tucker explains the problem. We need rules that say what is expected from a Parallel_Iterator.

Brad: !problem: "cursors return by" → "cursors returned by"

Approve AI with changes: 11-0-2. Arnaud, Ed abstain.

AI12-0362-1/01 Attributes for fixed point types

Jeff suggests we hold this one for potentially adding it later, as opposed to killing it by voting it No Action.

Hold AI: 11-0-2. Jeff, Brad abstain.

AI12-0362-2/01 Attributes for fixed point types

Gary says that !problem has a misspelling of “signifi{c}ant”. Jeff notes that a comma is needed between Floor and Ceiling.

“fixed[-]{ }point”. (This is in the both problem and the wording.)

Approve AI with changes: 12-0-1. Bob abstains.

AI12-0363-1/04 Fixes for Atomic and Volatile

Tucker explains Full_Access_Only and “full access object”.

In 3.10.2(26/3) “atomic object” should be “full access object”.

The GNAT Volatile_Full_Access aspect is intended to be the same as Volatile + Full_Access_Only.

There is a missing } at end of C.6(6.10/3).

C.6(12.1/5) “A corresponding rule applies to volatile types.” => “A corresponding rule applies to volatile types and similarly to full access types.”

Tucker will think about the implication of the generic array rule for full access types. [After the meeting, he sent the following: Add to the end of the formal array type paragraph “A corresponding rule applies when the actual type has volatile full access components.”]

Gary suggests that in C.6(13.3/5): “passed as {an}[the] actual parameter”. We agree to this after some discussion.

Tucker has typo in C.6(8.1/4): “Redundant” is misspelled.

In !discussion there are two misspellings: “address{a}[i]bility” “address{a}[i]ble”.

Steve wonders about generic private types used in a full access type. He thinks there needs to be an assume-the-worst for that in generic bodies.

Tucker doesn’t think it makes much sense to use a generic formal type in a full access objects. So he suggests adding to the end C.6(8.1/4) “or is of a generic formal type”. Add an AARM note to explain that the actual might be a full access object.

Approve AI with (extensive) changes: 10-0-2. Bob, Gary abstain.

AI12-0372-1/02 Static accessibility of “master of the call”

Tucker explains the new rule.

“return expression of an expression function” is a defined term.

Randy asks if we need to say something about a “generic function”. Tucker says bodies of generic and non-generic functions are both “function bodies”, so this rule covers both.

In the question: “anonynous” → “anonymous”.

In the discussion: “It can be necessar[il]y to...”.

Brad notes that “Delete 3.10.2(19.3/4)” should be “Replace 3.10.2(19.3/4)” and then “with”.

Approve AI with changes: 12-0-1. Gary abstains.

AI12-0377-1/02 View conversions and out parameters of types with Default_Value revisited

Tucker explains the changes.

This AI is misclassified, it should be a Binding Interpretation.

Brad notes a typo in !discussion: Move the closing paren to before the first period. Tucker would prefer to insert a semicolon before the open paren, and then drop the parens.

Mention AI12-0074-1 as a source of more examples of “deinitialization”.

Approve AI with changes: 12-0-1. Jeff abstains.

AI12-0378-1/03 View conversions of out parameters of access types revisited

This AI is misclassified, it should be a Binding Interpretation.

Steve asks if the accessibility check might be dynamic. Yes, it is similar to a membership check, usually the result is statically known but not always.

It is asked about the expense. Randy and Tucker note that silently changing the behavior of very old code that currently works would be bad. We need to change as little behavior as possible. Tucker thinks that this will usually be a static check.

Richard has typo in the 3rd paragraph of the discussion: “as usage sites” => “at usage sites”.

Tucker says “convertable” should be “convertible”. “incompatib{i}lity”

Steve asks if “ApexAda” is correct, we decide it is.

Ed suggests “side-steps” is “sidesteps”.

“not an even byte” => “not byte-aligned”.

Approve AI with changes: 5-0-8 Steve, Arnaud, Bob, Richard, Claire, Jeff, Gary, and Justin abstain.

This doesn’t pass (a majority of people voting need to be in favor). Most people say that we do not want runtime work to decide whether to pass **null**. We want a compile-time determination of whether to pass **null**.

Randy notes that whatever we do, this should be a Binding Interpretation.

The AI will go back to Tucker for another attempt.

AI12-0379-1/02 More presentation issues

Response (5) and (7) “it’s” should be “its”.

(9) “WG 9-approved” is weird, drop the space.

(7) has “stemps”.

(4) in !wording the paragraph number 90-1 should be written 90-91.

Approve AI with changes: 13-0-0.

AI12-0381-1/02 Tag of a delta aggregate

There is a build-in-place issue, and also a performance cost.

Steve does not like that it is different than 'Old, but it can't be helped with the build-in-place issue.

Gary says the !problem has an extra hyphen in "compile[-]{ }time"

Approve AI with change: 13-0-0.

AI12-0382-1/01 Loosen type-invariant overriding requirements of AI12-0042-1

Gary explains that this AI tightens up the rule so that it doesn't apply in cases where the subprograms are not boundary entities.

Brad suggests that the inheritance of Is_OK is should be commented like the other routines.

In the penultimate paragraph of !question, add a comma after "ancestor".

"be tightened up" → "be refined"

Approve AI with changes: 13-0-0.

AI12-0383-1/01 Renaming values

Jeff worries about unintended consequences. Randy notes that renaming a value is still a value.

Approve AI with changes: 12-0-1. Jeff abstains.

AI12-0385-1/01 Predefined shifts should be static

Gary would like to change the subject to add "and rotates".

Brad notes a typo in the !problem.

"That prevents [them from] using them"

Also in !problem, "shift{ing}" - We decide that "shifting and rotating" is how they are referred to in B.2.

Arnaud should propose an AI to describe allowing importing of static intrinsic functions. (Any such thing would be implementation-defined.)

Approve AI with changes: 13-0-0.