Minutes of Electronic ARG Meeting 62Q

23 June 2022

Attendees: Steve Baird, John Barnes (left at 13:20), Randy Brukardt, Arnaud Charlet (joins at 11:03), Jeff Cousins, Gary Dismukes, Claire Dross, Bob Duff, Edward Fish, Brad Moore, Ed Schonberg, Justin Squirek, Tucker Taft, Tullio Vardanega, Richard Wai (joins at 11:01, left at 12:32).

Observers: None.

Meeting Summary

The meeting convened on Thursday, 23 June 2022 at 10:34 hours EDT and adjourned at 13:29 hours EDT. The meeting was held using Zoom. The meeting covered much of the agenda.

AI Summary

The following AI was approved:

AI22-0042-1/01 Freezing rule needed for dispatching expression functions (14-0-1)

The following AIs were approved with editorial changes:

AI22-0019-1/02 Freezing of aspect specifications (14-0-1)
AI22-0027-1/03 Primitive equality for class-wide type (15-0-0)
AI22-0029-1/03 External_Tag collisions (14-0-0)
AI22-0037-1/01 Freezing of static expressions (14-0-1)
AI22-0038-1/03 Introduction for Type Invariants (13-0-0)
AI22-0041-1/02 Deferred constant subtype compatibility (15-0-0)
AI22-0044-1/01 Aggregate discriminants that do not belong to their subtypes (14-0-1)
AI22-0046-1/01 Statically names misses some cases (15-0-0)

The intention of the following AI was approved but it requires a rewrite:

AI22-0009-1/01 Nonvolatile views of volatile objects (13-0-1)

The following AIs were discussed and assigned to an editor:

AI22-0024-1/01 Abstract prefixed views
AI22-0036-1/02 Attributes in the expressions of Default_Value
AI22-0045-1/01 Issues with pragma placements

Detailed Minutes

Welcome

Steve welcomes everyone.

Apologies

Jean-Pierre Rosen has a conflict with a training session. Richard Wai has conflicts with other meetings and will both be late and will have to leave early. Arnaud Charlet has another meeting and will join late.

Previous Meeting Minutes

There were no comments on the minutes: Approve minutes: 13-0-0.

Date and Venue of the Next Meeting

Randy proposes Thursday, September 8, 2022 for our next meeting.

John says it is his Diamond wedding anniversary. We congratulate him and invite him to spend it with us.

There are no objections to the September 8th date; so we adopt that date. The meeting time and duration remains the same.

FDIS Progress Report

Randy reports that draft 33 of the Reference Manual, with all of the FDIS rewordings and reorganizations applied, was posted early Tuesday morning. An FDIS was made available to WG 9 as well.

There will be a short review period in order to ensure that no significant errors (like leaving out a clause) occur in either document. Given the substantial reorganization, there is some risk of that happening (although the RM was compared to a previous version to check that unintended changes did not occur). That will close on July 5th.

A final Ada 2022 version of the RM will be created and posted after the ISO publication date is known.

Steve thanks Randy and Tucker for getting all of this work done.

Working Group Report – UCI (User Community Input)

Since Richard is not present, Tucker makes a short report. We are now considering the Github and Google Docs directory as the main source working locations. We've improved the directions and more as these have been used. Randy notes that he is adding cross-references between the AIs and the Github issues (in both directions) and labeling the Github issues appropriately.

Later, when Richard joined, we ask him if he has anything to add to the previous report (which, of course, he didn't hear). He says that the Working Group is declaring victory. (By this he means that the WG is not planning any further meetings and that the individual members will finish up their action items – Editor.) The sites are up and running.

Randy notes that he's given Tucker an action item to propose a rewording of the "**Instructions for Comment Submission"** portion of the Introduction in the Reference Manual. We probably don't want to be sending people to the Ada-Comment list anymore, if that is not the primary method of feedback.

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.

Steve Baird:

- AI22-0009-1
- AI22-0036-1

Randy Brukardt:

- AI22-0019-1 (create an example of the problem)

Editorial changes only:

- AI22-0027-1
- AI22-0029-1
- AI22-0037-1
- AI22-0038-1
- AI22-0041-1

- AI22-0044-1
- AI22-0046-1

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. (or choose another, untested area)

Tucker Taft:

- AI12-0346-1 – also, construct the Technical Report suggested by this AI.
- AI22-0024-1
- AI22-0034-1
- AI22-0045-1
- ACATS C-Test(s) for container aggregates
- Propose a rewording for the RM **Instructions for Comment Submission** to reflect the new UCI processes.

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 2022 AARM, the number refers to the text in draft 33 of the Ada 2022 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 2022 AIs

AI22-0009-1/01 Nonvolatile views of volatile objects

Steve explains the problems.

Tucker doesn't like the one-way rule for value conversions. We try to wordsmith wording that would require a copy if the types do not exactly match vis-a-vis volatility and atomicity. (But those two words are not Ada terms, so we can't use them in the wording.)

Tucker wondered about conversion about unrelated array types where one has volatile components. Randy had thought that was illegal, but it is illegal only for view conversions (so it is allowed for value conversions, which is the case in question). So we need to cover such cases.

Tucker notes that all elementary view conversions are new copies, so we're only worrying about composite objects in this new rule. Therefore, he suggests using "volatile part" in the rule. "Corresponding parts need to be volatile."

Steve takes the AI back for another attempt to reword.

Approve intent: 13-0-1. John abstains.

AI22-0019-1/02 Freezing of aspect specifications

Steve would like an example in the AI of a problem with the current wording. Randy is volunteered to create such an example and have Steve and Tucker check it.

The new wording only freezes aspects that are evaluated as part of freezing.

Gary would like shorter wording “shall not cause freezing” rather than “shall not be one that causes…”.

Approve AI with changes: 14-0-1. Jeff abstains.

AI22-0024-1/01 Abstract prefixed views

We discuss this in detail. We would like the obvious case to work. Ideally, a prefixed view is equivalent to the similar explicit call. Randy notes that some of these cases have no counterpart elsewhere in Ada, as one has a curried subprogram that carries over some additional requirements on the remaining parameters.

Tucker thinks that an equivalence rule will work. He will take the AI.

Keep alive: 13-0-1. John abstains.

AI22-0027-1/03 Primitive equality for class-wide type

There is an unmarked redundant bracket in the first fix for 3.4.1(5): (But that’s later deleted.)

After discussing the two possible fixes, we ask which fix everyone prefers. No one speaks in favor of the simple fix, so we will use the better fix. Remove the simple fix from the AI.

There is a stray “note that this” in the !discussion.

Add “Editor’s Note: “ in the square bracketed text before the Chapter 12 changes.

Approve AI with changes: 15-0-0.

AI22-0029-1/03 External_Tag collisions

Tucker explains the issue and the rewording.

Brad notes a double question mark in the !issue.

Approve AI with changes: 14-0-0.

AI22-0036-1/02 Attributes in the expressions of Default_Value

Brad notes that Default_Value for a character type cannot be set to any value that does not have a literal (like Nul). This proposal is intended to eliminate that problem.

Steve would prefer that we simply say that the current instance rule does not apply to the prefix of certain attributes in the expression of Default_Value. Then it would be treated as a type, and the usual attribute definitions would apply. It is agreed this is preferable.

There is a brief discussion as to whether Pos would be useful in this context. Ed Fish suggests that T’Val(T’Pos(Nul) +1) could be useful in some contexts (to get the literal after a particular one). The author will consider that.

Steve will take the AI for a rewrite.

Keep alive: 12-0-1. Gary abstains.

AI22-0037-1/01 Freezing of static expressions

Randy notes that this is a ramification; no language change is proposed. He originally thought that one was needed, but determined in writing it up that he was wrong. He thought it was important to keep that reasoning for future reference, especially since it was already written.

In the discussion, remove case (2) and the renumber the others.

Brad notes in !ACATS test: “Note that {there} are already”.

Gary in the same paragraph, there is a missing closing paren.

!discussion, “whereever” should be “wherever”.

Steve complains that this does not answer his original question. Randy says that the answer to that question is not obvious since the equivalence rule cannot be considered to be equivalent in all cases (since conformance must not match, particularly in the array case – see the !appendix mail). And it is very difficult to construct a case where any difference in freezing would matter, so the importance of answering the question is low. He should create an AI if he wants an answer.

Approve AI with changes: 14-0-1. Jeff abstains.

AI22-0038-1/03 Introduction for Type Invariants

Tucker adds “either” after “postcondition”.

Ed Fish suggests a rewording that Tucker didn’t like.

Randy notes that there are other reasons for enforcement than to two given in this introduction. “Either” makes it sound like these are the only ones. Tucker suggests changing “either” to “including”.

We then start asking whether we need the last comma. Eventually we decide the part after the comma only belongs to the last item and thus the comma isn’t needed (and is confusing).

Steve suggests: “The type invariant is enforced at certain points, including …” It’s not shortening anything, so this suggestion is dropped.

Approve AI with changes: 13-0-0.

AI22-0041-1/02 Deferred constant subtype compatibility

Steve explains that we don’t want a deferred constant completed with a value that doesn’t meet the (visible) predicate. Thus we add a new requirement that the full constant has a subtype that is statically compatible with that of the deferred constant. Ignoring predicates, we have that or more as a requirement already. The existing rule about exclusions becomes redundant with this change, so it is removed.

Tucker would like the wording instructions changed to “Replace” and “with”.

Approve AI with changes: 15-0-0.

AI22-0042-1/01 Freezing rule needed for dispatching expression functions

A primitive expression function can be called by a dispatching call, so it needs to be frozen when that can happen.

We use a rule very similar to the existing rule for ‘Access to accomplish this. Arnaud notes that building a dispatch table is very much like taking ‘Access of a subprogram, so it makes sense to treat these the same way.

Tucker notes than an expression function is never a completion. An expression_function_declaration can be a completion, but that is not subject to this (and other) rules; it acts much like a body.

Approve AI: 14-0-1. Jeff abstains.

AI22-0044-1/01 Aggregate discriminants that do not belong to their subtypes

Tucker notes that "belongs" does not include predicates, so the !issue and !recommendation needs to also mention satisfy the predicates. The wording already does that, so it doesn't need to be changed. [Editor's note: the subject needs to be changed as well.]

[After the vote on this AI, Brad e-mailed an additional typo, the ACATS test section ends with a double period.]

Approve AI with changes: 14-0-1. Gary abstains.

AI22-0045-1/01 Issues with pragma placements

Randy explains all of the problems (there are four of them).

How to proceed? Tucker notes that for things like "list(on)" and "warnings(on)", one probably doesn't want a lot of restrictions in terms of placement.

Ed Fish suggests that we want to categorize the pragmas, and potentially allow the different categories of pragmas in different places.

Executable pragmas should probably be heavily restricted.

What are the locations that GNAT allows for pragmas?

We turn to the fourth issue. Should we allow pragmas in declare expressions? Claire notes that they needed an assertion in a declare expression. Executable pragmas don't seem like a pragma. We should allow those. It's not as clear for other pragmas.

Richard notes that we then could have a pragma within a pragma:

```
pragma Assert ((declare
                   pragma List(On);
                begin True));
```

Perhaps it should be implementation-defined whether a pragma can be placed into a declare expression. Tucker suggests perhaps we should simply say that implementations can ignore non-executable pragmas in declare expressions (hopefully with a warning).

Brad notes a typo, which Tucker fixes.

Tucker will take this AI and make a full proposal.

Keep alive: 15-0-0.

AI22-0046-1/01 Statically names misses some cases

We look briefly at the lengthier refactoring of the wording, but the need to introduce a third term is not appealing, so we stick with the duplicative fix.

The inserted text should not start with "it" (that's part of the lead-in). Also, "occuring" is misspelled.

Approve AI with changes: 15-0-0.