Minutes of Electronic ARG Meeting 62B

11 March 2020

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

Observers: Arnaud Charlet.

Meeting Summary

The meeting convened on Wednesday, 11 March 2020 at 11:02 hours EDT and adjourned at 14:01 hours EDT. The meeting was held using Zoom. The meeting covered many of the AIs on the agenda.

AI Summary

The following AIs were approved:

AI12-0347-1/04 Presentation issues (13-0-1)
AI12-0370-1/01 Pattern to use for specifying a precondition (12-0-2)

The following AIs were approved with editorial changes:

AI12-0079-3/03 Global-in and global-out annotations (11-0-3)
AI12-0350-1/01 Swap for Indefinite_Holders (14-0-0)
AI12-0359-1/01 Calls to subprograms declared in shared passive units (14-0-0)
AI12-0361-1/01 Ada.Streams.Storage packages are still useful (14-0-0)
AI12-0364-1/01 Add a modular atomic arithmetic package (14-0-0)
AI12-0367-1/01 Glitches in aspect specifications (14-0-0)
AI12-0368-1/02 Declare expression can be static (14-0-0)
AI12-0369-1/01 Relaxing barrier restrictions (14-0-0)
AI12-0371-1/01 Fix-ups for aspects in generic formal parts (13-0-1)

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

AI12-0282-1/04 Atomic, Volatile, and Independent generics formal types (14-0-0)
AI12-0372-1/01 Static accessibility of “master of the call” (12-0-2)

The following AI was discussed and assigned to an editor:

AI12-0363-1/01 Fixes for Atomic

Detailed Minutes

Welcome

Steve welcomes everyone.

Apologies

Bob Duff is in the process of moving, and might be late or have a bad connection. No one knows why Gary Dismukes isn't here. (Shortly after we mentioned this, both Bob and Gary joined the meeting.)

Previous Meeting Minutes

No one has any changes to the minutes (all previously sent suggestions have already been applied). Approve minutes: 14-0-0.

Date and Venue of the Next Meeting

Our next face-to-face meeting will be June 12-14 in Santander, Spain immediately after the Ada-Europe conference.

We briefly discuss the COVID-19 pandemic, which has the potential to disrupt travel into June and beyond. (Coincidentally, this was the day when WHO finally officially declared a pandemic.) We agree that we would try to have some sort of lengthy remote meeting during those dates if an in-person meeting is not possible (if Ada-Europe is canceled, for instance). We'll certainly have remote call-in even if the in-person meeting is held. [It won't be; ISO officially canceled all in-person meetings

through the end of June the next day.]

AdaCore Feedback

Arnaud says that he would like to give a brief overview of the result of the review of Ada 202x AIs made by AdaCore's LD Circle. He apologizes for not mentioning it sooner so it could have been added to the agenda.

The LD Circle has completed reviewing all of the AIs. The group is happy with the majority of the AIs. But they could not get a consensus on some AIs, AdaCore has decided to not implement some AIs at this stage. They believe that these AIs would benefit from more prototyping and experimentation: especially parallel programming (don't think that GPU programming is sufficiently supported). That includes AI12-0119-1, AI12-0251-1, AI12-0266-1, and AI12-0242-1. AI12-0079-3 is still in process (to be discussed today). They also do not like the procedural iterator proposal (AI12-0189-1, AI12-0286-1). He mentions that they don't like the exit mechanism which seems to involve a special exception; there's also some sentiment that a more general mechanism would make sense. Finally, AI12-0262-1 and AI12-0242-1 (map/reduce) – they think a library approach would make more sense.

Randy asks about Nonblocking (since it wasn't mentioned). Arnaud says that it is on the fence; it has uses beyond the parallelism mechanisms.

Jean-Pierre asks whether it makes sense to include those AIs in the Standard. A Standard that no one is planning to implement is dubious at best. Randy suggests moving them to a Technical Specification, we don't want to lose the overall work, and that way they would be available for implementers if/when demand picks up. Tucker would like to defer this discussion until later, as no one has had a chance to think about this. We agree to take up this subject at our next meeting.

Unfinished Action Items

There are three unfinished action items (Steve Baird, AI12-0016-1; Bob Duff, AI12-0243-1; Ed Fish: AI12-0215-1). 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
- AI12-0243-1 (help Bob Duff on request)

Randy Brukardt:

Editorial changes only:

- AI12-0079-3
- AI12-0350-1
- AI12-0359-1
- AI12-0361-1
- AI12-0364-1
- AI12-0367-1
- AI12-0368-1
- AI12-0369-1
- AI12-0371-1

Bob Duff:

- AI12-0243-1 (with help from Steve Baird)

Edward Fish:

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

Tucker Taft:

- AI12-0282-1
- AI12-0363-1
- AI12-0372-1
- Help the Editor convert language-defined preconditions to the form recommended by AI12-0370-1.
- Help the Editor update the containers Global aspects (see AI12-0079-3).

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 24 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

AI12-0079-3/01 Global-in and global-out annotations

This is close to what is supported by SPARK, other than syntax (“in” vs. “input”), “synchronized”, and we don't have “state abstractions”.

SPARK doesn't allow specifying all of the visible objects in a package, so that capability has been dropped. They have an aversion to having multiple ways to name an object. Public children are not included in the state of their parent package.

Specifying a default is still allowed, but mainly for library-level packages. A nested routine has to follow the Global of its enclosing subprogram, but otherwise can do up-level addressing.

The model of generics was changed as well. Generics now follow the "Pure" model - an instance is pure if the actuals are pure. Applied to Globals, this means that the generic has the union of the Globals of its formals (and any inherited Global).

Allowing Globals to be explicitly "unspecified" allows compatibility with SPARK's "peek at the body" approach.

Steve asks Claire about the SPARK perspective on this proposal. She thinks that it works for now, there are not a lot of conflicts. She thinks that deferring the access type rules is the right solution for now.

Tucker continues through the Annex H paragraphs.

Randy asks about constants that are mutable. Tucker finds the rule, and notes that there is an Implementation Permission to ignore constants that the compiler can tell are immutable but are otherwise subject to the rule (this generally means a minor privacy breakage).

Steve asks where the check that an access-to-subprogram is compatible with the Global. Tucker says there needs to be such a Legality Rule.

Brad wonders if “unspecified” should be “unchecked”. Tucker says that it's not unchecked; the compiler can reject an “unspecified” use if the subprogram actually violates the usage.

Brad has 6 typos. We ask that he sends those to the ARG list and we'll apply them.

Approve with changes: 11-0-3. Abstain: Ed, Gary, and Jean-Pierre.

Jeff would like an example. We should replace the TBD by the one used in the !wording. Randy asks Tucker for help updating the containers (which is the example).

AI12-0282-1/04 Atomic, Volatile, and Independent generics formal types

There is a serious incompatibility with these rules, and that was not discussed when the AI was approved.

If you have a generic formal private type and one has an actual type that is atomic, that would be made illegal by this AI. Generic formal private types are used for “universal” uses.

Someone asks Randy about his objection to reverting the change. He says that Tucker convinced him that a program that cared precisely how many atomic reads and writes happened would be unusual (nearly pathological). All of the reads and writes would be atomic, there would just be a slightly different number in some cases where a copy is required by the language but it wouldn't happen. Given all of the other race conditions possible, this seems like a very unlikely concern.

We note that the AI has separate rules for requiring the formal to match the actual, and actual to match the formal, which is silly and confusing.

There is no change here, that needs to be proposed yet.

Tucker volunteers to take this, and proposes to reword to improve the presentation, and remove the requirement for an exact match for most formal types (it remains for formal derived types).

Approve intent: 14-0-0.

AI12-0347-1/04 Presentation issues

Randy notes that there are 4 minor changes.

Brad Moore asks if we should put both specs into the E.4.2 example. Randy wanted the minimum change.

Approve AI: 13-0-1. Bob abstains.

AI12-0350-1/01 Swap for Indefinite_Holders

Randy explains that a Swap operation can be performed for an Indefinite_Holder by simply swapping the internal pointers; an extra object is needed if we don't have this operation. It doesn't help as much for a Bounded_Indefinite_Holder as a copy is required, but the extra object is still saved and we like keeping the two kinds consistent.

The parens should be curly brackets in the wording: “Move {and Swap} should not...”

The precondition should be written in the “or else” form.

Approve AI with changes: 14-0-0.

AI12-0359-1/01 Calls to subprograms declared in shared passive units

Richard Wai asks if Microsoft Windows is the appropriate example. Randy thought that "shared libraries" is rather generic, and DLLs are well known. Tucker agrees with Randy, even a Linux-centric person such as himself knows what a DLL is.

Arnaud notes that this isn't going to affect any implementations, since they are not working on Annex E anymore.

Brad Moore notes a typo, an extra square bracket in the editor's note.

Say "of" instead of "in" in the last sentence of the AARM Ramification.

Approve AI with changes: 14-0-0.

AI12-0361-1/01 Ada.Streams.Storage packages are still useful

Tucker explains how the package works. Tucker says he's implemented this several times. Randy notes that Ada has Storage_IO, but this is more useful since there is no control over the representation of Storage_IO.

Typos: "jusification".

Extra "should" in the second paragraph of the !problem.

Approve AI with changes: 14-0-0.

AI12-0363-1/01 Fixes for Atomic

The first and third questions have fixes. Randy thought that the second question did not need a change.

Tucker wonders if we need the last sentence of C.6(13.3/5) at all. Randy tries to explain the purpose of the rule.

Tucker and Randy argue about the need for the last sentence of C.6(13.3/5). No conclusion is reached. Everyone else wants to move on.

We decide to not take a vote, since the intended change is unknown.

Tucker will take the AI.

AI12-0364-1/01 Add a modular atomic arithmetic package

Brad suggests in the !problem "an index {in}to a ring buffer".

There is a question whether we should describe this as the same as the integer one, similarly to the way that Wide_Text_IO is described in terms of Text_IO. Randy thinks this seems not long enough for that to be necessary.

Should we rename the signed package? Jean-Pierre suggests following Integer_IO and Modular_IO.

Does anyone object to changing the existing package name to Integer_Arithmetic? Straw poll:

Arnaud doesn't want to change it. Justin will agree with Arnaud. Everyone else votes for the change. 13-2-0.

So change to Integer_Arithmetic.

Approve AI with changes: 14-0-0.

AI12-0367-1/01 Glitches in aspect specifications

Gary notes "e{n}umeration" in the wording.

Approve AI with changes: 14-0-0.

AI12-0368-1/02 Declare expression can be static

Randy and Tucker explain the AI.

Brad has a typo in !discussion: "(semantically) contents" should be "contents (semantically)".

Approve AI with changes: 14-0-0.

AI12-0369-1/01 Relaxing barrier restrictions

Tucker explains the problem. Changing "statically denoted" to "statically named" solves this. The real-time people seemed OK with this change.

Tucker notes that "Jorvik" is misspelled.

Gary points out that in the !problem:

```
Count : Natural :;
```

should be:

```
Count : Natural := 0;
```

The Rec definition is messed up:

```
type Rec is record
   Flag  : Boolean;
   ... <other stuff> ...
end record;
```

should be:

```
type Rec is record
   Flag  : Boolean := False;
   Count : Natural := 0;
end record
   with Dynamic_Predicate => (Count /= 0) or not Flag;
```

Approve AI with changes: 14-0-0.

AI12-0370-1/01 Pattern to use for specifying a precondition

Randy notes that the direct translation of the wording of the containers gives "if".

Tucker again volunteers to help converting the hundreds of preconditions in the containers.

Approve AI: 12-0-2. Abstain: Jeff, Brad.

AI12-0371-1/01 Fix-ups for aspects in generic formal parts

Brad notes that the workaround is wrong,

```
procedure Actual is Proc;
```

should be

```
procedure Actual renames Proc;
```

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

AI12-0372-1/01 Static accessibility of “master of the call”

Tucker does not like the second sentence of the wording. Randy notes that the mechanism to convert one “master of a call” to a second “master of a call” which then applies this rule is never specified in the Standard. But that is required to make the rule work for Steve's example.

Tucker would like to take this off-line to reword. Randy says that he had hoped that would have already been done (he provided this AI to Steve and Tucker on Monday). But obviously it was not.

Gary notes a typo at the end of the !question, “anonymous” is misspelled.

Brad wonders about “specifically don't”; this is a plural form for a single paragraph reference. There are two paragraphs here, maybe adding “paragraphs” to the front would clarify that.

Approve intent: 12-0-2. Jean-Pierre and Gary abstain.