Notes on telephone ARG meeting (#43A), March 17, 2011 1:00 PM EDT

Attendees: Steve Baird, Randy Brukardt, Robert Dewar, Gary Dismukes, Bob Duff, Brad Moore, Ed Schonberg, Tucker Taft.

Summary of Action Items (finish by April 1, 2011):

Tucker:

- AI05-0115-1: Check for and fix any problems with existing uses of "descendant".
- AI05-0183-1: Study issues with static expressions and premature freezing, and work out a solution. Add the ability to have aspect_specifications on subprogram bodies. Add a blanket rule allowing implementations to do whatever they want (even for syntax) for implementation-defined aspect_specifications.
- AI05-0197-1: Determine how Eiffel handles merging of Pre/Post when different ones are inherited.
- AI05-0229-1: Reword pragmas Import/Export/Convention into aspects, and propose Annex J wording for the pragmas.

Randy:

- AI05-0197-1: Split contract aspect issues into a new AI.
- AI05-0229-1: Reword remaining pragmas into aspects, and propose Annex J wording for the pragmas.
- AI05-0243-1: Figure out appropriate wording.

Bob:

- AI05-0245-1: Send comments on proposed wording, read the rest of the Introduction for possible changes.

Ed:

- AI05-0215-1: Change By_Any to Optional; make other corrections noted during the call.
- AI05-0229-1: Help with conversion of pragmas to aspects (exact ones TBD).

Steve:

- AI05-0110-1: Polish the wording.
- AI05-0229-1: Reword Preelaborable_Initialization into an aspect, and propose Annex J wording for the pragma.

The following AIs were Approved (sometimes with changes):

- AI05-0139-2/09 Syntactic sugar for accessors, containers, and iterators (8-0-0)
- AI05-0197-1/02 Dispatching when there are multiple inherited operations (4-0-4)
- AI05-0200-1/01 Mismatches in formal package declarations (8-0-0)

- AI05-0206-1/03 Remote_Types and Remote_Call_Interface packages should be able to depend on Preelaborated units (7-0-1)
- AI05-0212-1/08 Accessors and Iterators for Ada.Containers (8-0-0)
- AI05-0218-1/02 Generics and volatility (8-0-0)
- AI05-0236-1/02 Additional Problem with preelaborated generics (8-0-0)

The following AI was voted No Action:

- AI05-0125-1/03 Nonoverridable operations of an ancestor (7-0-1)

Start: 1:10 PM EDT

Handling of Decisions

After the meeting, we'll circulate a list of decisions, and ask if anyone would like to question one of them. If so, they can call for a Letter Ballot.

AI05-0245-1/01 2011-03-10 -- Introduction update for Ada 2012

Bob has some comments, he will send the via e-mail. Bob will read the rest of the Introduction to see if there is any changes. We'll vote on this one next time.

AI05-0236-1/02 2011-03-15 -- Additional Problem with preelaborated generics

Approve AI: 8-0-0.

AI05-0139-2/09 2011-03-16 -- Syntactic sugar for accessors, containers, and iterators

Ed notes that they don't have any implementation experience with the indexing and reference aspects, but have implemented the iterators proposal.

Approve AI: 8-0-0.

AI05-0212-1/08 2011-03-16 -- Accessors and Iterators for Ada.Containers

Tucker has lost interest in the inner derived Set container, it would seem to make the containers too complex.

In the example, Randy forgot to change the for loop to:

```
for N in Node loop
```

`G.Element (Node)` should simply be `G (Node)` (using the new indexing features)

Capitalize the example title: "Example of Container Use"

Adjecency should be Adjacency in the last loop example.

Should the description text at the be a Note? Robert says no, an entire example is like a note. And there is agreement that this text is valuable, especially for newer users of Ada. (And who else reads the examples??) So leave it as it is.

Approve AI with changes: 8-0-0.

AI05-0183-1/09 2011-03-16 -- Aspect Specifications

Tucker will work on the staticness problem; he apologizes for missing it in his homework list.

Randy has suggested allowing aspect_specifications on subprogram_bodys. That seems "obvious". Tucker notes that it is annoying to separate this important information into a separate specification for main subprograms – it could easily get lost.

Should we allow anything at all for implementation-defined aspects? Specifically, any syntax? This doesn't seem to be a problem so long as we are not trying to ignore unrecognized aspects; we've decided that we don't want to do that. Randy notes that this would allow prototyping of the proposed, now dead global in/global out annotations and of exception contract annotations. Tucker will write some wording to allow anything goes for implementation-defined aspects.

Tucker explains the changes he made to the AI.

Typos will be e-mailed.

Approve intent: 8-0-0.

Deadline for updates: April 1, 2011.

AI05-0229-1/03 2011-03-14 -- Specifiable aspects

Randy notes that he was crushed by an avalanche of homework and a lack of promised help, so these didn't get close to done. But he did finish four aspects, so we can look at the wording for them.

13.2(1):

"If the Pack aspect is True, then storage minimization..."

Tucker notes that No_Return uses "Specifying".
He would like to see: "Specifying aspect Pack to have the value True indicates..."

Bob would prefer to change the No_Return to be like the first item above.

Randy and Ed prefer the one that starts with "if"; Tucker wants the other one but seems to have no one else that agrees with him. We'll use the "If" wording.

Robert worries that No_Obsolescent_Features will make the existing programs incompatible. He thinks

that would prevent some customers from moving to Ada 2012. Randy suggests exempting these pragmas from No_Obsolescent_Features. We already do that for a few such features. After some discussion, we agree to do that.

13.2(5):

"Aspect Pack is of type Boolean."

Deadline April 1.

Ed will take some of these (Randy will give him a list), Steve still has Preelaborable_Initialize, Tucker has Import/Export/Convention, Randy will take the rest.

Approve Intent: 6-1-1.

Robert Dewar is against obsolescing the pragmas, as a lot of useless work (which he says that he doesn't have to do, so this is not a strong objection). Ed notes that we want the RM to show the preferred way of doing things, which is with aspects.

AI05-0215-1/04 2011-03-17 -- Pragma Implemented should be an aspect (and better defined)

Tucker notes that his original suggestion of True doesn't make any sense. Jean-Pierre's suggestion of "Optional" seems like the best idea so far.

Robert wonders if we need the value at all. It defines a default value, and the wording refers to that often; it would be harder to write the wording without it.

Do we want a pragma? No.

Paragraph 9.5(18/3): "aspect Implemented" => "aspect Is_Synchronized" (several places).

Approve intent: 8-0-0.

AI05-0167-1/05 2011-03-15 -- Managing affinities for programs executing on multiprocessors

Drop "language-defined" from this wording Dispatching_Domain. This leads to a meta-discussion of removing it everywhere.

Randy wonders if we should drop the "may be specified with an aspect_specification" from everywhere. We think that "with an aspect_specification" is redundant (that mostly affects AI05-0229-1, but some others as well).

Pragma Dispatching_Domain needs to be born obsolescent. That seems politically expedient for Dispatching_Domain and CPU.

This was previously approved; reapprove with changes: 8-0-0.

AI05-0190-1/10 2011-03-16 -- Global storage pool controls

Bob explains the changes. The new bullet is confusing.

For the type of an access result, at the place where the accessibility level is determined by the point of call (see 3.10.2).

We decide to postpone this one to the end of the agenda (after AI05-0234-1). [But we run out of time before we get back to this one.]

AI05-0206-1/03 2011-03-16 -- Remote_Types and Remote_Call_Interface packages should be able to depend on Preelaborated units

Brad just added a paragraph at the end of the discussion to address the question that Steve raised at the last meeting.

Robert says that he cannot think of an implementation problem. Randy notes that you can do this in the body, just not the private part.

Tucker "mentioned {in}[by]..." in several places.

Approve AI with changes: 7-0-1.

AI05-0111-3/07 2011-01-28 -- Subpools, allocators, and control of finalization

Should we change anything here?

Robert says that Brad's proposal arrived long after the deadline. And it is clearly not a slam dunk; there are problems that would need to be worked out. So we should simply thank him for the contribution but not consider it.

Should we remove the current proposal? The main proponents (especially Tucker) give a strong no. This is functionality that currently is not available, and a less than perfect solution is way better than no solution. So we will keep the current AI as is.

AI05-0125-1/03 2011-02-09 -- Nonoverridable operations of an ancestor

Tucker explains his position. This is the case of giving a declaration where the "grandparent" operation is not inherited, but you have visibility on it.

The current situation is that if you give such an operation with no overriding indicator, you get a new operation. If you say "overriding", you get an error.

The change if nothing is said, you get an error; if you say "overriding" it works.

The problem with interfaces is that you get a preference rule; and makes interfaces funny.

We thought about making these illegal, because they are effectively privacy breaking. But that causes a ripple effect – anything declared in the private part could break existing code.

Tucker thinks this one is rapidly approaching the "brick wall of complexity". Randy says that his opinion on this one flops around like a fish caught in a gill net.

So long users actually use overriding indicators, they will at least be aware of the problem. (That was the main driving force for adding them in the first place.) They'll get an error if they expect overriding but don't get it. It might require heavy restructuring (which is bad), but there will not be surprises. So this is not worth crashing into a wall of complexity.

No Action: 7-0-1.

AI05-0197-1/02 2010-03-17 -- Dispatching when there are multiple inherited operations

Steve explains that null procedure that aren't equivalent is a problem. That happens with 'Class contracts.

Randy suggests that we split this problem (with Pre'Class/Post'Class/Type_Invariant'Class) to discuss this more generally.

The problem is how conflicting Pre/Post/Invariants are dealt with, especially in the case of multiple inheritance.

The question was raised as to what Eiffel does in this case. Tucker and Bob will research.

Approve AI with changes (to remove discussion of this issue): 4-0-4.

Tucker and Bob disagree on how Eiffel works; they will take that off-line.

AI05-0243-1/03 2011-03-09 -- Clarification of categorizations

Randy needs to fix. "Declared pure" has to stay (we're not going to change 37 uses). But "declared pure library unit" doesn't include a limited view. Perhaps use "declared pure library_item". He will get this done.

AI05-0200-1/01 2010-10-21 -- Mismatches in formal package declarations

Ed explains his changes.

Gary does not like "shall be a legal set of actuals". That is weird. Steve suggests "legal sequence of actuals".

Bob: "The generic actual parameters of the generic formal package associations shall be legal for an instantiation of the template."

Ed: "If all of the formal_package_associations are given by generic associations, the generic_actual_parameters of the formal_package_associations shall be legal actuals for an instantiation of the template."

Bob will e-mail the rewrite of this sentence.

Approve AI with changes: 8-0-0.

AI05-0110-1/02 2011-03-03 -- Inheritance of characteristics of generic formal derived types

Tucker wants predefined operators to be a characteristic. Randy says that there are separate rules for that, we don't want to introduce bugs in this area. Tucker notes that predefined operators are not inherited primitive subprograms. But the wording in 3.4(7) is OK, because it says "implicitly-declared", not "inherited". Fix the AARM note to say "primitive subprograms (including predefined operators)...".

The idea is that this list of bullets in 3.4 is the definition of what characteristics are; we want to make sure that it is clear which things are characteristics and which are not.

7.3(16), add "user-defined" in front of primitive subprograms (twice). This is putting this back. Also do that in 12.5.1(20/2) (two more places).

The old text for 12.5.1(21/2) has some extra wording in it, which needs to be deleted.

The wording in 12.5.1(20/2) only talks about inherited things; predefined operators are not inherited.

Tucker suggests: "Characteristics of a formal derived type, it's predefined operators, and it's inherited primitive subprograms are determined by its ancestor type and progentitors (if any), in the same way the characteristics, predefined operators, and primitives subprograms are determined for a record extension by the ancestor type and progenitors."

This could use some polishing. Steve will take this one back to fix this wording.

Approve intent: 8-0-0.

AI05-0218-1/02 2011-03-03 -- Generics and volatility

Last time, we discovered that formal array types have the same problem as formal derived types. So this was updated to handle the formal array case.

Robert notices that a packed array cannot be Volatile because it would not allow an extra read which is necessary. That seems like bug. Volatility mostly implies that it can change externally. There needs to be a permission to read before write for Volatile. Make an Ada 2020 Binding Interpretation. We need to make it clear that a Load can occur at the point of a Store.

Approve AI: 8-0-0.

AI05-0115-1/06 2011-03-16 -- Aggregates with components that are not visible

Tucker explains that "descendent" depends on views. Else it would be a problem with legality rules. It is likely that any changes would fix bugs in the standard rather than cause them. But Randy reminds everyone that the first use he looked at calculated this at run-time, which surely should not depend on views.

Tucker will check if there is a problem for "descendent". By April 1st.

Approve intent: 8-0-0.

Next Phone Meeting

We probably will want another phone meeting to clean up last minute issues. April 7, 2011 1 PM EDT.

Stop: 4:10 PM EDT