Subject: Comments on Clause 12
From: Van Snyder

1 Edits — and comments without editorial suggestions

Edits refer to 06-007. Page and line numbers are displayed in the margin. Absent other instructions, a page and line number or line number range implies all of the indicated text is to be replaced by associated text, while a page and line number followed by + (-) indicates that associated text is to be inserted after (before) the indicated line. Remarks are noted in the margin, or appear between [and] in the text.

[Editor: "or" ⇒ comma, insert ", the appearance of an object processed by user-defined derived-type input/output (9.5.3.7) in an input/output list, or finalization (4.5.6)" at the end of the sentence.] 297:21-22

[Wouldn't it be clearer if internal subprograms were constrained against appearing within internal subprograms at [11:51+]?] 297:34-298:1

[Internal subprograms are now allowed to be actual arguments. Editor: Delete ", the internal procedure name shall not be argument associated with a dummy procedure (12.5.1.6)".] 298:3-4

[Editor: Add an item to the list:] 300:1-4

(z) because an object processed by user-defined derived-type input/output (9.5.3.7) appears in an input/output list,

---

[It seems that the only reason an interface body can specify that the procedure is not pure is if the procedure itself is defined to be pure. Editor: Insert "even" before "if".] 302:25

[Editor: For consistency with [302:12-14] "An explicit . . . way" ⇒ "If an external procedure does not exist in the program, an interface body for it may be used to specify an explicit specific interface but the procedure shall not be used in any other way".] 302:27-29

[The note seems to say that dummy arguments specified in a procedure definition or an interface body might be the same as other dummy arguments in the same definition or interface body. Editor: "may be different" ⇒ "in an interface body may be different from the corresponding dummy argument names in the procedure definition".] 302:30+2

[Subclause 16.3.4 doesn't have anything to do with scope of local identifiers, which is the topic of Subclause 16.3. It more logically belongs here anyway. Editor: Move Subclause 16.3.4 here. Do we thereby need to convert some of the ordinary normative text in it to constraints?] 304:14+

[What's the point of the "nonoptional" restriction? It just prevents a class of defined-assignment routines to be used for other purposes. Editor: "arguments. . . . nonoptional" ⇒ arguments that are".] 305:13-14

["the right-hand side enclosed in parentheses" could, if taken literally, cause two copies if the second argument has the VALUE attribute. VALUE doesn't preclude INTENT(IN), and there's nothing here that precludes VALUE. Editor: "INTENT(IN)" ⇒ "the INTENT(IN) or VALUE attribute", insert "actual" between "first" and "argument", "enclosed . . . second" ⇒ ", enclosed in parenthses if the second dummy argument does not have the VALUE attribute, as the second actual".] 305:15,18,19

[Editor: Newline needed between R1215 and R1216.] 307:10

[Editor: "initial-proc-target" ⇒ "*initial-proc-target*".] 307:11

[Editor: "initialization target" ⇒ "**initialization target**". `\tdef` will add it to the index, which is needed, since the only one there refers to [65:26] and has to do only with variables.] 307:44

[Feature creep: Editor: "or module" ⇒ "module, or internal", insert "If the initialization target is an internal procedure, it shall be an internal procedure of the main program." at the end of the paragraph.] 307:44-308:1

[Editor: Would be clearer at [308:5+].] 308:9-10

[Editor: Insert ", by the appearance of an object processed by user-defined derived-type input/output (9.5.3.7) in an input/output list, or by finalization (4.5.6)" at the end of the sentence.] 310:1

[At [311:13-16] one can conclude that no more than one actual argument can correspond to a dummy argument. Editor: "actual arguments that correspond" ⇒ "an actual argument that corresponds".] 317:5+2

[Doesn't work for optional dummy arguments that don't correspond to actual arguments. Editor: "actual ... or" ⇒ "associated actual argument shall be a function, or the corresponding actual argument, if any, shall be a".] 318:20

[Doesn't work for optional dummy arguments that don't correspond to actual arguments. Editor: "actual ... or" ⇒ "associated actual argument shall be a subroutine, or the corresponding actual argument, if any, shall be a".] 318:22

[Editor: "associated ... argument" ⇒ "present".] 319:40

[I can't find the normative justification for the "but not both" part, at least not within the list in which Note 12.32 is embedded, not least because the actual argument has the POINTER attribute, while the list refers to allocation status (which only applies to allocatable variables).] 321:0+5

[Editor: Insert "of" after "value".] 321:4

[Editor: Delete "or", insert ", or finalization" after "item".] 322:16-17

[Editor: For consistency with 8.2 and the discussion of termination of execution of constructs "control may be transfered" ⇒ "a branch may occur", "statements" ⇒ "branch targets".] 322:18

[Editor: for consistency with [323:11], delete second "if".] 323:21

[Editor: Simplify the paragraph by replacing it:] 327:28-32

The *prefix-spec* RECURSIVE shall appear if any function defined by the subprogram directly or indirectly invokes any function defined by the subprogram.

NOTE 12.40a

Each ENTRY statement in the subprogram defines an additional function.

[But don't do both this and section 2 below.]

[Simplify. Editor: ", the pointer ... disassociated" ⇒ "its pointer association status shall not be undefined".] 327:41-42

[Editor: Simplify the paragraph by replacing it:] 329:19-23

The *prefix-spec* RECURSIVE shall appear if any subroutine defined by the subprogram directly or indirectly invokes any subroutine defined by the subprogram.

NOTE 12.40a

Each ENTRY statement in the subprogram defines an additional subroutine.

[But don't do both this and section 2 below.]

[Editor: "function or subroutine" ⇒ "procedure".] 329:29

Each instance ceases to exist when execution of the invocation that created the instance completes exection, or execution of the program is terminated. 329:38+ New ¶

[Editor: "keyword" ⇒ "prefix" twice.] 330:3-4

["Is accessible" is sometimes interpreted to mean "is available by use or host association." Editor: "is only accessible by use association" ⇒ "can be accessed by use association only".] 330:22+2

[There is no point to constrain an *entry-stmt* against appearing within an *executable-construct*, since the 331:1-2 syntax doesn't admit it. Editor: delete "An *entry-stmt* shall not appear within an *executable-construct*." (or replace "shall" by "cannot" and make the sentence a note).]

331:31,33-35 [We usually use "executable construct" instead of "executable statement". Editor: "statement" ⇒ "construct". An ASSOCIATE statement is executable. Can the name of a dummy argument in an ENTRY statement appear as the *associate-name* in a prior ASSOCIATE statement? It ought to be allowed, since the *construct-name* is the name of a construct entity. Editor: Insert "it is the name of a statement or construct entity (16.4), or it" after "unless". Does this need to get into a Fortran 2003 corrigendum? Editor: Insert "as a dummy argument name" after "appears".]

[Editor: Replace [331:33-35] by the following note:

NOTE 12.45a

If a prior statement or construct entity has the same name as a dummy argument in an ENTRY statement, and its type and type parameters are determined by the scoping unit in which it appears, it will either be implicitly typed or it will necessarily appear as a dummy argument in a prior SUBROUTINE, FUNCTION or ENTRY statement.

332:6 [Editor: Insert "or another procedure defined by the subprogram" after "itself".]

332:19-21 [Editor: Insert "of the procedure name referenced" after "list". For consistency with 8.2 and the discussion of termination of execution of constructs "transfers control" ⇒ "branches", insert "of the procedure name referenced when the RETURN statement is executed" after "specifier in the argument list", "transfer of control" ⇒ "branch".]

333:7 [Editor: Either delete "an" or insert "function" after the first "intrinsic", insert "function" after the second "intrinsic".]

333:24 [Editor: "the entity" ⇒ "an entity", insert ", or has type and type parameters determined by the implicit typing rules in effect in that scoping unit" after "function".]

333:28 [Editor: "attributes" ⇒ "parameters".]

333:33-34 [Wouldn't it be simpler to say "a pure intrinsic procdure"? Compare to [335:12]. If it works there, why not here?]

334:7 [Editor: Insert ", or within the *specification-part* of a BLOCK construct within a pure subprogram," before "shall". In light of C1281, delete "or *internal-subprogram-part*".]

334:25+3+ [Why is there a page break here? There's no `\newpage` in sight.]

335:2 [Editor: "assignment" ⇒ "defined assignment , user-defined derived-type input/output". Yes, DTIO can happen, so long as the unit is an internal file.]

335:8 [Could this be folded into C1282?]

336:8+1-3 [Repairing the note to account for IMPURE would result in it saying "the constraints on pure procedures apply to pure procedures." Editor: Delete Note 12.51.]

2 Reorganization of "Procedures defined by subprogams"

Subclause **12.6.2 Procedures defined by subprograms** repeats material from suclauses 2.3.4 and 12.6.2.3 (which it cites), so it's not needed. On the other hand, this is a good place to collect the material about the *prefix*.

325:32-34 One or more procedures are defined by each subprogram. In addition to the procedure entry point defined by the initial SUBROUTINE or FUNCTION statement, an additional procedure entry point is defined by each ENTRY statement (12.6.2.5).

A procedure is specified to be elemental (12.8), pure (12.7), recursive, or a separate module procedure (12.6.2.4) by a prefix of its initial SUBROUTINE or FUNCTION statement.

[Editor: Move [326:25-327:10] to here.]

The *prefix-spec* RECURSIVE shall appear if any procedure defined by the subprogram directly or indirectly invokes itself or any other procedure defined by the subprogram.

[Editor: Move [328:1-4] to here.]

[Editor: Delete [327:28-32] and [329:19-27].]