Subject: Comments on Clause 4
From: Van Snyder

1 Edits

Edits refer to 06-007r1. 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.

[Get rid of “may,” which might be ambiguously interpreted as “permission” or “possibility.” Editor: “may . . . real” ⇒ “for logical and bits is completely determined by this part of ISO/IEC 1539. The set of valid values for integer, character and real is processor dependent.”. Make definitions for complex and derived type parallel: “For complex values the set of valid values” ⇒ “The set of valid values for complex”, “For derived types . . . as” ⇒ “The set of valid values for derived types is”.]] 41:12-16

[“may” is probably OK here, but if it’s not, “may” ⇒ “can” twice.] 41:20-21

[Get rid of “may,” which might be ambiguously interpreted as “permission” or “possibility.” Editor: replace the paragraph.] 41:22-23

A constant value of derived type is constructed from an appropriate sequence of initialization expressions (7.1.7) by a structure constructor (4.5.10). Such a constant value is a scalar even if it has array components.

[Arrays are conspicuously absent. Editor: Insert a new paragraph:] 41:23+

A constant array value is constructed from an appropriate sequence of initialization expressions (7.1.7) by an array constructor (4.7).

[“may” is probably OK here, but if it’s not, “may” ⇒ “can” twice.] 41:29,32

[“may” is probably OK here, but if it’s not, “may” ⇒ “can” thrice.] 42:1,4,8

[“may” is probably wrong here because there are no other possibilities once one has decided to have a type parameter. Editor: “may be” ⇒ “is’. Cannonball polishing: “with” ⇒ “by”.] 42:11

[In case “may” isn’t OK here, “may“ ⇒ “shall not”, “only” ⇒ “except”.] 42:16

[The phrase “it is also used. . . ” is ugly and misleading: Is “it” the assumed type parameter, the length type parameter, or the dummy argument? Editor: “; it is also . . . for” ⇒ “, a length type parameter for an associating entity that assumes the type parameter value from the corresponding selector in a SELECT TYPE construct, or”. Since the length type parameters of associating entities in ASSOCIATE constructs are not declared, add “The length type parameters of an associating entity in an ASSOCIATE construct are considered to be assumed type parameters.” at the end of the paragraph.] 43:2-5

[Apply the Weaver rule: Say something twice, say it wrong at least once. The proper place for the description of assignment is subclause 7.4. Extensive discussion of assignment is especially inappropriate here given that it is only “one means of defining or redefining. . . .” Editor: Insert “(7.4)” after “Assignment”, then delete “Assignment . . . (7.4.1.4)” at [43:16-20] and Note 4.4.] 43:15-20+2

[Everything after the first sentence duplicates syntax rules. Note 4.5 at [44:5+1-3] is enough of a reminder. Editor: Delete “in . . . used.” Otherwise, delete Note 4.5.] 43:24-26

[Doesn’t work for BLOCK constructs. Editor: Insert “in the specification part of a program unit” after “*declaration-type-spec*”.] 43:35

[Given C403 at [43:29], C406 is not needed. Editor: Delete C406.] 44:10

[Certainly a polymorphic pointer entity can also be allocated to be of a type-compatible type. Editor: Insert “or pointer” after “allocatable”.] 45:1

[C417 unnecessarily restricts how character length can be specified in ALLOCATE statements or array constructors. It further duplicates C404 at [43:34-36], which imposes the correct constraint. Editor: Delete C417.] 50:17

[Editor: Delete "which represent" since the two values *are* true and false.] 53:5

[What trailing delimiter? Editor: Delete "following the trailing delimiter".] 53:17

[The "bits" feature added .XOR. as another nonequivalence operator. Editor: Insert "or .XOR." after ".NEQV.".] 53:20

[Earlier, we are careful to say "previously defined or accessed by use or host association." Editor: To avoid the impression that the parent type has to be defined in the same scoping unit "a previously . . . (4.5.7)" ⇒ "an extensible type (4.5.7) that is previously defined or accessed by use or host association".] 56:1

[Editor: Insert "(4.5.7)" after "extended type" because type extension is a forward reference.] 60:5

[Earlier, we are careful to say "previously defined or accessed by use or host association." Editor: To avoid the impression that the component type has to be defined in the same scoping unit " previously . . . type" ⇒ "derived type that is previously defined or accessed by use or host association".] 61:8-9

[Needs to say that the components are finalized in a processor-dependent order. Editor: "Each . . . finalized" ⇒ "All finalizable components that appear in the type definition are finalized in processor-dependent order"] 71:6

[The syntax term *generic-spec* includes *dtio-generic-spec*. Since [74:1-2] and [74:3-5] say the same thing, the latter isn't needed, but in some cases it does have more precise words. Editor: Insert "type-bound" before "generic" and "for *generic-spec*" after "interface" at [74:2], then delete [74:3-5].] 74:1-5

[Doesn't account for use or host association. Shouldn't this be a constraint?] 76:14

The *derived-type-name* in the *derived-type-spec* in a structure constructor shall be the name of a previously-defined type or a type accessible by host or use association.

[Don't say "rank-one" again since it's already said at [79:2]. Editor: Delete "rank-one".] 80:17

2 Questions without edits

Constraint C416 makes it pretty pointless to allow the kind of the length type parameter of character entities to be processor dependent. Which way do we want it? 50:16

I don't understand C471. 68:25-26

I don't understand why item (2) doesn't work like item (1). 71:6-8

Should this be "When a procedure is invoked an actual argument that is associated with a nonpointer nonallocatable INTENT(OUT) dummy argument is finalized"? 71:33-34