To: /B subgroup
Subject: Comments on section 8
From: Van Snyder

1 Edits

Edits refer to 01-007r2. 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 immediately following text, while a page and line number followed by + (-) indicates that immediately following text is to be inserted after (before) the indicated line. Remarks are noted in the margin, or appear between [and] in the text.

[The ASSOCIATE and SELECT TYPE constructs aren't in the list, and the list isn't in alphabetical order.] 145:7-9

(1) ASSOCIATE construct
(2) CASE construct
(3) DO construct
(4) IF construct
(5) SELECT TYPE construct

[The sentence "Unless there is a transfer of control out of the block, the execution of the block is completed when the last executable construct in the sequence is executed" could be construed to say that a transfer out of the block does not complete the block's execution. Editor "Unless ... executed" ⇒ "Execution of the block is completed when the last executable construct in the sequence is executed or when a transfer of coutrol out of the block takes place".] 146:6-8

[The "will be" at [146:35] is not parallel to the "is" at [146:34], and probably isn't proper standardese. Editor: "will be" ⇒ "is".] 146:35

[It is not clear why the CONTINUE statement is referenced here. Editor: Delete "and execution continues as though a CONTINUE statement (8.3) were executed".] 147:34

[Editor: "is present" ⇒ "appears". The term "present" applies to optional arguments.] 148:44

[Editor: "is absent" ⇒ "does not appear". The term "absent" applies to optional arguments.] 148:46

[Editor: after "type" insert "that is an extension of the declared type of the *selector*".] 151:2

[The term "associating entity" is not specifically defined. There's no edit here, just an observation.] 151:36-42

[Editor: "the those" ⇒ "those".] 151:44

[The phrase "may not" can be read "it perhaps wouldn't happen" (although such reading is perhaps prohibited by the rules of standardese). Editor: To remove the possibillity of this confusion, "may not" ⇒ "is prohibited to".] 152:34

[Editor: "with" ⇒ "by".] 155:11

[Editor: insert "of" before "m_3".] 156:22

[The sentence "When a DO construct becomes inactive, the DO variable, if any, of the DO construct retains its last defined value" is nearly, but not precisely identical to "When an active 157:43-44

DO construct becomes inactive, the *do-variable*, if any, retains its last defined value" at [156:7-8]. Although in this case, the difference doesn't lead to a misinterpretation, there's no point to repeating it. Editor: Delete.]

[Editor: Insert ", an *associate-stmt*, an *end-associate-stmt*" after "*action-stmt*". Insert ", a *select-type-stmt*, an *end-select-type-stmt*" after "*end-select-stmt*".] 158:43-44

[Doesn't prohibit anything, and doesn't allow anything that would otherwise be prohibited. Editor: Make it a note.] 159:16

[Doesn't prohibit anything, and doesn't allow anything that would otherwise be prohibited. Editor: Make it a note.] 159:25

[Editor: Insert a blank before "(14)".] 159:39

1.1 Problems with branching

These problems are spread across several pages, so the discussion can't be coherently put into page-and-line order with the above discussions.

The first problem is that the prohibitions against branching to the end statement of a construct are duplicated. An alternative to the solution implied here is to concentrate the prohibitions against branching to the end statement of a construct except from within the construct in 8.2. If that is the preferred solution, one might also argue that the body of 8.1.1.2 belongs in 8.2.

The second problem is that there is no discussion of what statements cause branching.

[Editor: "the" ⇒ "its" (see [158:47]).] 147:2

[Editor: "the" ⇒ "its" (see [158:46]).] 149:5

[Editor: "the" ⇒ "its".] 152:6

[Editor: "the" ⇒ "its".] 152:27

Branching may be caused by a GO TO statement, a computed GO TO statement, an arithmetic IF statement, a CALL statement that contains an *alt-return-spec*, or an input/output statement that has an END= or ERR= specifier. Although procedure references and control constructs can cause transfer of control, they are not branches. 158:42+ No new ¶

[These statements almost duplicate [147:1-2], [149:4-5] (where it says "the" instead of "its"), [155:39-41] (where it says "that" instead of "its") and [155:44-1] (where the wording is entirely different, including the word "corresponding," which isn't here). There's also nothing here about SELECT TYPE or ASSOCIATE constructs. Duplication is not useful, and it's especially dangerous when it's not exact. Editor: Delete.] 158:46-2

1.2 Related material in section 16

[Editor: "with" ⇒ "to" for consistency with the rest of the subclause.] 377:8

2 Question

We need to be more precise about the scope of the association of an *associate-name* to its *selector*.

It isn't clear, either from 8.1.4.4 or from 16.7.1.5, whether the *associate-name* is associated to the *selector* during execution of the ASSOCIATE statement. It would be convenient to be able to write

```
associate ( x => table(i), y => x%element(j), z > y%component(k) )
```

Do we wish this to work? If so, we need to say something to the effect that the scope of the association of an *associate-name* to its *selector* extends from immediately after the *association* until the *end-associate-stmt*. (We don't want it to start too early, because we don't want to get stuck in a loop by `x` => x%y.) Otherwise we need to say that its scope is the *block*. The discussion of association is presently spread between 8.1.4.4 and 16.7.1.5, so it's not obvious where to put this.