To: /B subgroup
Subject: Comments on Section 12
From: Van Snyder
References: 01-271

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.

[Internal procedures also differ from external procedures by always having explicit interface. Editor: delete "and" at [237:28]; insert ", and internal procedures always have explicit interface" after "association" at [237:29].]	237:28-29
[Could be read to say that a subprogram defines a procedure only if it has an ENTRY statement. Editor: "If ... it" $\Rightarrow$ "A subprogram" at [237:31]; insert ", if any," before "and" at [237:32].]	237:31-32
[The phrase "whether it is a pointer or a target" could be read to require that it be one or the other. Editor: delete "and"; "or" $\Rightarrow$ ", and whether it is".]	238:18
[Assumed size is not obsolescent. Editor: Set size in the normal font.]	238:22
[The phrase "whether it is allocatable or a pointer" could be read to require that it be one or the other. Editor:"or" $\Rightarrow$ ", whether it is".]	238:30
[Editor: Insert a comma after "both".]	239:39
[Duplicates [240:33]. Editor: Delete the constraint.]	240:21
[Editor: Move [230:33-38] to here (after making changes indicated for that region below).]	240:23+
[Editor: Insert "it" before "shall".]	240:38
[Editor: "R1202" $\Rightarrow$ "R1206".]	240:43
[The reference to "procedure declaration statement" is forward. Editor: Insert "(12.3.2.3)" after "statement".]	241:19
[Editor: Insert "and it is accessible" after "one".]	242:47
[Editor: "on" $\Rightarrow$ "for".]	242:49
[The sentence of which this line is a part is a run-on sentence. Editor: "and" "$\Rightarrow$" ";".]	243:32
[Number doesn't agree. Editor: "an interface" "$\Rightarrow$" "interfaces".]	245:13
[Subclause 16.1.2.3, in a section entitled "Scope, association and definition," is not an obvious place to look for the requirements on the relation between the several interfaces in a generic interface block.. Editor: Move 16.1.2.3 to here, making it $12.3.2.1.3\frac{1}{2}$, i.e. one level lower than it is now, and calling it "Unambiguous generic interface declarations". Notice that there are edits for 16.1.2.3 [367:34,35] described below. You probably want to do them first.]	245:13+
[There is nothing that says what abstract interfaces are used for.]	245:16+

NOTE 12.10$\frac{1}{2}$

> Abstract interfaces are used in the declaration of procedure pointers and external procedures (12.3.2.3).

[Most references to "section" are capitalized. Editor: Capitalize "section".] 245:41

[The term "present" refers to optional dummy arguments. Editor "is present" ⇒ "appears".] 246:30

[Editor "be present" ⇒ "appear".] 246:31

[Constraint C1219 at [246:31-33] appears to need "and vice-versa":] 246:33+

C1219$\frac{1}{2}$ (R1211) If *proc-interface* appears and the *abstract-interface-name* it specifies is declared with a *language-binding-spec*, a *language-binding-spec* shall appear in the *proc-declaration-stmt*.

[Editor "is present" ⇒ "appears" twice.] 246:34,37

[The term "absent" refers to optional dummy arguments. Editor "is absent" ⇒ "does not appear".] 246:41

[Editor: Delete "accessible".] 247:23

[Editor: Insert "(15.2.3)" after "module"; "then" ⇒ "required to be".] 247:24

[The term "in the declared type" (used at [248:11]) is not defined. Editor: Add a new sentence in the same paragraph: "A binding is **in a type** if it is declared within the type definition, or it is inherited (4.5.3.1) into the type."] 248:13

[The term "corresponding specific interface" is not defined.] 248:21+

A specific interface in a dynamic binding **corresponds** to a specific interface in a declared binding if it is the same interface (either because the declared and dynamic types are the same or the interface is inherited into the dynamic type), or it overrides an inherited binding that corresponds to the specific interface in the declared binding. New ¶

[I don't think we need put the above instance of "corresponds" into the index.]

[Editor: Insert ", a procedure pointer" after "procedure".] 248:40

[Subclause 16.1.2.4, in a section entitled "Scope, association and definition," is not an obvious place to look for the requirements and methods for procedure reference, which is the topic of the present subclause. Editor: Move 16.1.2.4, except for 16.1.2.4.4, to here, making it 12.4.$\frac{1}{2}$ (i.e. one level higher than it is now). 16.1.2.4.4 is moved to Section 9 by paper 158-wvs-16. Notice that there are edits for 16.1.2.4 below [369:24,30]. You probably want to do them first.] 249:21+

[Editor "is present" ⇒ "appears".] 249:26

[Editor: "an allocatable or" ⇒ "allocatable or a" twice.] 250:24,25

[Editor: Insert "and" before "is"; delete "that".] 250:25

[Editor: Move to [252:5+]. Also see remarks below in section 2.] 250:33-38

[Duplicates note 5.16 at [75:1-22] exactly.] 252:23-44

NOTE 12.25

> Argument intent specifications serve several purposes. See note 5.16.

[There is no reason to put these constraints away from the syntax rules to which they apply. 253:1-7

Editor: Delete [253:1] and move [253:2-7] to [249:7+].]

[Editor: "function procedure pointer, or" ⇒ "a function procedure pointer, a reference to a function that returns a function procedure pointer, or a".] 253:33

[Editor: "subroutine procedure pointer, or" ⇒ "a subroutine procedure pointer, a reference to a function that returns a subroutine procedure pointer, or a".] 253:36

[Editor: Insert "nonoptional" before "nonallocatable".] 254:41

[A note about an arcane point wouldn't hurt:] 254:43+

NOTE 12.28$\frac{1}{2}$

> The requirement that a dummy data object that is not present shall not be referenced includes the case that it appears in a specification expression in the declaration of another dummy argument that is present.

[These paragraphs could be read to say that a dummy argument that has the POINTER attribute, or has the TARGET attribute but not INTENT(IN), cannot be used. Editor: "Action ... unless" ⇒ "Actions that affect the value of the entity or any subobject of it may access the entity or its subobjects directly if" at [255:4-5]; add a new paragraph as part of item (1) at [255:10+]:] 255:3-10

> Otherwise, actions that affect the value of the entity or any subobject of it shall be taken only through the dummy argument.

[Editor: "an intent of IN" ⇒ "INTENT(IN)".] 256:26

[Editor: For consistency with [255:4-5], "of any part of the entity" ⇒ "the entity or any subobject of it".] 256:32

[These paragraphs could be read to say that a dummy argument that has the POINTER attribute, or has the TARGET attribute but not INTENT(IN), cannot be used. Editor: "only ... unless" ⇒ "directly if" at [256:34]; add a new paragraph as part of item (2) at [256:39+]:] 256:29-39

> Otherwise, actions that affect the value of the entity or any subobject of it shall be taken only through the dummy argument.

[It isn't quite correct that X may be referenced after execution of INIT is complete. Editor: Insert "in the main program" before "once". We could add much more verbiage about "other subroutines that don't have the same problems as INIT" but this is, after all, only an example.] 257:16

[Editor: Insert "or any *entry-name* in the same subprogram" after "*function-name*".] 258:18

[Editor: "is an asterisk" ⇒ "is an asterisk".] 258:40

[Editor: "be present" ⇒ "appear".] 259:10

[Duplicates [262:1-2], which is the right place for it. Editor: Delete this one.] 259:12

[Editor: "is present" ⇒ "appears".] 259:14

[Editor: Move [259:30-48] to here (after making changes indicated for that region below, and answering questions about that region posed in section 2 below).] 259:15+

[Editor: "be present" ⇒ "appear" twice.] 259:25-27

[A conjunction ought not be used to begin a sentence. Editor: "However, because" ⇒ "Because"; insert ", however," before "used".] 259:45

[Editor: "is present" ⇒ "appears" twice.]	260:1,3
[There are two problems here: (1) It is no longer necessary for the function to be recursive for it to have explicit interface within itself, and (2) the paragraph duplicates [239:8-9], which is the proper place for it. Editor: Delete it (but if you must keep it, "both ... are" ⇒ "RESULT is").]	260:5-6
[Editor: "be present" ⇒ "appear".]	261:6
[Duplicates [262:1-2], which is the right place for it. Editor: Delete this one.]	261:8
[Editor: "is present" ⇒ "appears".]	261:10
[Editor: Move [261:18] to here.]	261:11+
[Editor: "be present" ⇒ "appear" twice.]	261:12,14
[Duplicates [239:8-9], which is the proper place for it. Editor: Delete this one (but maybe not, if you decide to keep the one at [260:5-6]).]	261:17
[Editor: "is present" ⇒ "appears" twice.]	261:19-21
[Editor: "be present" ⇒ "appear".]	262:3
[Editor: "and ... EXTERNAL" ⇒ "nor shall it appear in an EXTERNAL, PROCEDURE".]	262:6
[Editor: After "*entry-name*" insert ", the *entry-name* in another *entry-stmt* or the *function-name*".]	262:8
[Duplicates [239:8-9], which is the proper place for it. Editor: Delete this paragraph (but maybe not, if you decide to keep the one at [260:5-6]).]	262:20-21
[Duplicates [239:8-9], which is the proper place for it. Editor: Delete this paragraph (but maybe not, if you decide to keep the one at [260:5-6]).]	262:25-26
[Editor: "program" ⇒ "subprogram".]	262:29
[The phrase "and it is present (12.4.1.6)" is covered in 12.4.1.6 and 7.1.6 (C709 and item 2). By saying it here, it appears to imply that it need not be present in other circumstances. Editor: Delete it.]	262:44
[Editor: "is present" ⇒ "appears".]	263:18
[Editor: If the advice in 01-264 concerning the definition of what constitutes a branch is not taken, insert the following note:] **NOTE 12.40$\frac{1}{2}$** A transfer that results from an alternate return is a branch.	263:21+
[Editor: After "statement" insert "that does not have a *proc-language-binding-spec*".]	263:38
[BIND is no longer a *prefix-spec*. Editor: "the BIND *prefix-spec*" ⇒ "a *proc-language-binding-spec*".]	263:38-39
[Editor: Insert "*proc-*" immediately before "*language-binding-spec*".]	264:44
[Editor: "singal" ⇒ "signal".]	265:11
[Editor: "to" ⇒ "of".]	265:20

[Editor: Insert "or procedure pointer arguments" after "arguments".]	266:17
[Editor: Insert "or procedure pointer" after "procedure".]	266:18
[Editor: "*block*" ⇒ "*body*".]	266:24
[Editor: Insert ", if" before "any"; "with" ⇒ "has".]	266:26
[Editor: Insert "is" before "an".]	266:28
[Editor: Insert "it" before "shall".]	266:29
[Editor: Insert "*wait-stmt*," before "*backspace-stmt*".]	266:41
[Editor: Before "or" insert "that does not have an *io-unit*, or a *read-stmt*".]	266:43
[Editor: "used" ⇒ "possible". Then close up some of the empty space before the next paragraph (if possible).]	267:16
[Editor: "be present" ⇒ "appear".]	267:31
[Editor: "to" ⇒ "of".]	268:4
[How else than by a generic name or a specific name could an elemental procedure be referenced? Do defined operations count? Editor: "If a generic name" ⇒ Regardless whether a *generic-spec*".]	268:27
[Editor: "betwen" ⇒ "between".]	367:34
[Editor: "acesible" ⇒ "accessible".]	367:35
[Editor: "block" ⇒ "body" twice.]	369:24,30

2 Don't know what to do (or too lazy to figure it out)

There are no edits in this section.

This constraint allows, by omission, that one could access an interface by use or host association, and then put a specific procedure into that interface that is already in it. Is that the intent?	240:44-45
Does this allow or prohibit the case that a procedure has several names (because of renaming during USE) in a scoping unit?	241:23-24
Do we want to require that they all have the same result type as the derived type name?	243:5
What attributes, if any, are prohibited for dummy arguments of procedures that define operations or assignment? Are TARGET, VALUE and VOLATILE permitted? Is a procedure dummy argument or a procedure pointer dummy argument allowed?	12.3.2.1.2
Should VOLATILE be in the list?	246:11-16
I have no idea what the phrase "the name of the last argument shall be PRINT" does here.	250:14-15
This paragraph appears to be incomplete. It doesn't address INTENT(INOUT) or unspecified intent.	250:23-28
Neither this paragraph, nor any other that I could find in 12.4.1.2, address the case that the dummy argument has assumed length.	250:33-38
This can't work for rank > 1 without more explanation. Suppose we have an array of rank 2, with N rows. Don't we need the dummy argument to have a multiple of N elements? If not, we need to say something to the effect that some of the last rows are not present in the last column (and shall not be accessed). I think it gets worse in higher rank, but I haven't given	254:18-19

much thought to the question.

The ", but not both" part is not accurate if there are intervening ALLOCATE statements.	255:47
The term "procedure heading" isn't defined. Is it the *function-stmt* or the *specification-part* or both?	259:23-24
Does this paragraph need to address allocatable function results?	259:31-41
Is the ALLOCATABLE attribute permitted?	262:17-19
The term "procedure header" isn't defined. Is it the *function-stmt* or *subroutine-stmt*, or the *specification-part*, or both?	259:23-24
A procedure having a BIND attribute contradicts [70:10] (but that's the topic of 01-271).	263:32
It's not clear why we need BINDNAME= and these paragraphs. If the binder name is derived from the name of the external procedure by the companion processor, and the processor knows which companion processor(s) is(are) to be used, it can derive the binder name(s). Why ask every user to do the research? Are C programmers asked to provide binder names?	264:18-31
The term "base object" appears to be defined only for derived types.	266:26
The term "base object" appears to be defined only for derived types.	268:3
The "has only scalar arguments" part duplicates [262:42].	268:41
Should this "must" be "shall"?	269:5