Subject: Syntax and Semantics of VALUE attribute, issue 214, part of issue 90
From: Van Snyder
References: 00-171

1 Introduction

I propose that the syntax for what is now called the VALUE attribute ought to be INTENT(VALUE), and that the semantics should be that the subprogram can change the dummy argument, but the associated actual argument is not thereby changed. This is the way it works in C if the asterisk is omitted from a formal argument.

The reasons for this proposal are

- The semantics of the VALUE attribute are more different than necessary from the semantics of omitting the asterisk from a dummy argument in C,
- The semantics of the VALUE attribute are less useful than they would be if they were more similar to the semantics of omitting the asterisk from a dummy argument in C, and
- The standard would be simpler if the syntax were INTENT(VALUE): Almost everywhere the VALUE attribute is mentioned, the INTENT attribute is mentioned in the same sentence. If INTENT(VALUE) were used, the prohibition against duplicate specification would remove the need for any discussion of the relation between VALUE and INTENT.

Malcolm says the semantic change proposed here doesn't work because it runs afoul of the association rules in 12.4, 14.6.1.1, and item 12 in 14.7.5. Malcolm's "easy fix" is to use INTENT(VALUE) instead of VALUE. Then, all of the constraints that say "... VALUE, INTENT..." can just say "... INTENT...", and everywhere that says "INTENT(IN) and/or VALUE" would say "INTENT(IN) and/or INTENT(VALUE)".

I've put my original proposal for the semantics of INTENT(VALUE) as a separate section, so it's written down in case somebody really likes it and figures out how to make it work, and put in Malcolm's idea instead. Everything else is the same either way. The only justification that remains is the third point above.

2 Edits

Edits refer to 00-007r1. 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 immediately following text, while a page and line number followed by + indicates that immediately following text is to be inserted after the indicated line. Remarks for the editor are noted in the margin, or appear between [and] in the text.

[Editor: Delete "5.2.12 VALUE statement".]	iii
[Editor: Delete the syntax rule for the VALUE attribute.]	64:7
[Editor: Delete ", VALUE," because INTENT covers it.]	64:29

[Editor: Delete. Re-stated by the edit at [71:19+] below.] 65:15-20

The constraint at [65:15-17] includes a prohibition against PARAMETER, which isn't needed because the PARAMETER attribute is prohibited for dummy arguments at [63:27], and a prohibition against DIMENSION, which is prohibited at [65:21]. So even if the change proposed here is not accepted, the prohibitions against PARAMETER and DIMENSION should be removed from the constraint at [65:15-17].

Note to J3

[Editor: Delete. There's no reason for it.] 65:21-22

[Editor: Replace "VALUE" by "INTENT(VALUE)"' twice.] 65:28, 31

or VALUE 71:10+

Constraint: If the INTENT(VALUE) attribute is specified for a dummy argument of a subprogram or interface body that has a *language-binding-spec* 71:19+

(1) The dummy argument shall be a scalar, and

(2) If the dummy argument is of character type, the length parameter shall be omitted or shall be specified by an initialization expression with the value one.

Constraint: If the INTENT(VALUE) attribute is specified the ALLOCATABLE, POINTER, or VOLATILE attribute shall not be specified.

There is no need to prohibit EXTERNAL because that's covered by the constraint at [71:11-12]. If the change in 00-171 is accepted, a conspiracy of the revised constraint and the constraint against POINTER serves.

Note to J3

The INTENT(VALUE) attribute implies the INTENT(IN) attribute. A processor may choose, however, to use different argument passing mechanisms for INTENT(IN) and INTENT(VALUE) dummy arguments. 71:38+

Note 5.11$\frac{1}{2}$

The name of the INTENT(VALUE) attribute is intended to be suggestive. Although the processor is not required to use pass-by-value for an argument with the INTENT(VALUE) attribute, that might be a possible implementation. If the INTENT(VALUE) attribute is specified for a dummy argument of a procedure or interface body that has a *language-binding-spec*, the processor shall use the same argument passing convention as the companion processor, which is often pass-by-value.

[Editor: Delete section 5.1.2.14, including unresolved issue 214.] 78:15-35

[Editor: Delete.] 82:31-33

[Editor: Replace "VALUE (5.1.2.14)" by INTENT(VALUE) (5.1.2.3)".] 244:16

[Editor: Insert ", whether it has the INTENT(VALUE) attribute" after "pointer".] 244:23

[Editor: Replace "VALUE" by "INTENT(VALUE)".] 245:21

[Editor: Replace "the VALUE attribute" with "INTENT(VALUE)".] 257:36

[Editor: Replace "the VALUE attribute" with "INTENT(VALUE)".] 258:7

[Editor: Replace "`VALUE`" by "`INTENT(VALUE)`".] 390:24

[Editor: Replace "the VALUE attribute" by "INTENT(VALUE)" twice.] 392:33,35

[Editor: Replace "The VALUE attribute" by "INTENT(VALUE)".] 393:8

[Editor: Replace "`VALUE`" by "`INTENT(VALUE)`)" twice.]	393:37,40
[Editor: Replace "the VALUE attribute" by "INTENT(VALUE)".]	393:46
[Editor: Delete "VALUE" from the index twice.	467

3 What I originally had in mind

These have the form of edits, but are no longer part of the proposal.

Constraint: If the INTENT(VALUE) attribute is specified the ALLOCATABLE or POINTER attribute shall not be specified.	71:19+
The INTENT(VALUE) attribute for a nonpointer dummy argument specifies that if the value of the dummy argument is changed or becomes undefined during execution of the procedure, the associated actual argument is not affected. The INTENT(VALUE) attribute for a pointer dummy argument specifies that if the association status of the pointer changes or becomes undefined during execution of the procedure, the pointer association status of the associated actual argument is not affected.	71:38+
The constraint at [65:15-17] originally included a prohibition against VOLATILE. If the semantics are changed as suggested in the edit for [71:38+], there isn't a problem with the VOLATILE attribute: It would apply to the dummy argument, not the associated actual argument. The same reasoning applies to the comments about ASYNCHRONOUS in issue 90 at [65:28-31].	*Note to J3*