From: Malcolm Cohen

Subject: The PROTECTED attribute.

# 1. Introduction

The PROTECTED attribute described by 00-307 is a good idea. Various other languages (object-oriented and otherwise) languages have this useful facility. Unfortunately, that paper does not provide an adequate exposition of the idea, and goes completely off the rails in the EDITS section, where the edits do not implement the specs and syntax described in that paper.

# 2. Specifications and Syntax

The PROTECTED attribute can be applied to variables and procedure pointers in modules. This attribute prevents modification of the entity from outside the module.

Note 1: For pointers, this means alteration of the pointer association status, i.e. the PROTECTED attribute functions like the INTENT(IN) attribute, but for use-associated variables not dummy variables.

Note 2: The PROTECTED attribute is only useful for PUBLIC entities.

# 3. Edits to 00-007r3

[10:47+] Insert " **or** *protected-stmt*".

{Add statement form of the attribute.}

[66:3+] Insert "**or** PROTECTED".

{Add attribute form.}

[67:37+] Insert constraints:

"Constraint: The PROTECTED attribute is permitted only in the specification part of a module.

Constraint: The PROTECTED attribute is permitted only for a procedure pointer or variable that is not in a common block.

Constraint: If the PROTECTED attribute is specified, the EXTERNAL, INTRINSIC, PARAMETER, or PRIVATE attribute shall not be specified.

Constraint: A nonpointer object with the PROTECTED attribute that is accessed by use association shall not appear in a variable definition context (14.7.7).

Constraint: A pointer with the PROTECTED attribute that is accessed by use association shall not appear as

(1) A *pointer-object* in a *pointer-assignment-stmt* or *nullify-stmt*,

(2) An *allocate-object* in an *allocate-stmt* or *deallocate-stmt*, or

(3) An actual argument in a reference to a procedure if the associated dummy argument is a pointer with the INTENT(OUT) or INTENT(INOUT) attribute."

{Obvious constraints.}

[81:21+] Insert new subclause

"5.1.2.16 PROTECTED attribute

The PROTECTED attribute for a variable specifies that

- if it is a nonpointer object it shall neither be defined nor become undefined, and
- if it is a pointer it shall not have its association status changed,

other than by execution of a statement within the module in which the object is defined. Furthermore, such a statement shall not modify the object through argument association unless procedure from which the object is being passed as an actual argument is also within that module.

If an object has the PROTECTED attribute, then all of its subobjects have the PROTECTED attribute."

[84:39+] Insert new subclause

"5.2.13a PROTECTED statement

The PROTECTED statement specifies the PROTECTED attribute (5.1.2.16) for a list of entities."

[364:39] Insert ", 5.1.2.16" after "5.1.2.3".