

Subject: Use ALLOCATABLE and POINTER attributes in generic resolution
From: Van Snyder
Reference: 03-258r1, section 2.3.2

1 **Number**

2 TBD

3 **Title**

4 Use ALLOCATABLE and POINTER attributes in generic resolution.

5 **Submitted By**

6 J3

7 **Status**

8 For consideration.

9 **Basic Functionality**

10 Use ALLOCATABLE and POINTER attributes in generic resolution.

11 **Rationale**

12 I have a generic interface `Allocate_Test` that allocates an object, tests the status, and prints an error
13 message if an error occurs. I cannot have a specific interface for a to-be-allocated argument that has
14 the `POINTER` attribute, and another for a to-be-allocated argument with the same type, kind type
15 parameters and rank that has the `ALLOCATABLE` attribute. So I need different generic names for
16 allocatable objects and pointer objects. If I change an object from pointer to allocatable or vice-versa, I
17 have to track down all of the `Allocate_Test` invocations for that variable and change them to the other
18 one. Avoiding this labor was one of the justifications for the generic facility.

19 **Estimated Impact**

20 Minor.

21 **Detailed Specification**

22 Allow the `POINTER` and `ALLOCATABLE` attributes to be used for generic resolution. If the only
23 difference between two specific interfaces is that one has neither the `POINTER` nor `ALLOCATABLE`
24 attribute for some argument, and the other one has one of those attributes for the corresponding ar-
25 gument, the interface is ambiguous. If one has the `POINTER` attribute and the corresponding one has
26 the `ALLOCATABLE` attribute the interface is not ambiguous, at least so far as that pair of specific
27 procedures is concerned.

28 If a dummy argument has the `POINTER` or `ALLOCATABLE` attribute, the corresponding actual argu-
29 ment is required to have the same attribute. Therefore this change would not invalidate any existing
30 program.

31 **History**