Subject: Create new types from existing types
From: Van Snyder
Reference: 03-258r1, section 1.3.2

Number

TBD

Title

Create new types from existing types.

Submitted By

J3

Status

For consideration.

Basic Functionality

Provide a mechanism to create new types from existing types, independently from type extension and component embedding. A new type should interoperate with C if the type from which it is derived interoperates, although some other conditions may need to be satisfied as well.

Rationale

Programmers who use other languages have found it useful to be able to create new types from existing types, by means other than type extension or component embedding. Some languages classify these entities as names for existing types. Others classify them as new types. The principles that justify strong typing in Fortran militate against classifying them as new names for existing types.

Estimated Impact

Small to moderate. Most of the changes will be in Section 4.

Detailed Specification

Provide a mechanism to create new types from existing types, independently from type extension and component embedding. A new type should interoperate with C if the type from which it is derived interoperates.

One possible syntax is TYPE, NEW [, *access-spec*] :: *new-type-spec* => *existing-type-spec*.

It should be possible to parameterize the new type if the type from which it is derived is parameterized. If some of the parameters of the *existing-type-spec* are given values that are not simply names of parameters from the *new-type-spec*, the new type has fewer parameters than the base type. For example, suppose the existing type E has three parameters, and a definition for a new type N appears:

TYPE, NEW :: N(P) => E(5,P,KIND(0.0D0))

Then N is a new type with one parameter. This reduction in the parameters of E is similar to what is called "partial application" in functional programming languages.

History