Date: 26 June 1998
To: J3
From: Van Snyder
Subject: Discussion paper – Explicitly typed expressions and dummy arguments
References: 98-124, 98-146, 98-160, 98-171, 98-172

Paper 98-124 advocated explicitly typed array constructors. Papers 98-146, 98-160 and 98-172 advocated explicitly typed allocations. Paper 98-171 indirectly advocated allowing function result types to participate in generic resolution.

All of these facilities could be brought together in a consistent way by allowing an explicit type declaration to precede a primary in an expression. This facility has utility even when using function result types for generic resolution. For example if one has the two enumeration types used in paper 98-171:

```
TYPE :: COLOR => UNORDERED(RED, GREEN, BLUE)
TYPE :: NAMES => UNORDERED(WHITE, BROWN, BLACK, GREEN, BLUE)
```

and some functions with the generic name `FUN` that take arguments of types `COLOR` and `NAMES`, the reference `FUN ( GREEN )` is ambiguous, but `FUN ( TYPE(COLOR) :: GREEN )` is not.

The reasoning in the standard about type conversions in expressions could be spelled out in terms of facilities within the language. E.g.

`0.5d0 + 3` becomes

`0.5d0 + REAL(KIND(0.5d0)) :: 3` becomes

`0.5d0 + REAL(KIND(0.5d0)) :: REAL(3)` becomes

`0.5d0 + REAL(KIND(0.5d0))(3)`

Some people prefer to define everything about a name in one place. This is almost possible, except in the case of dummy argument declarations. Allowing an entire type declaration, including attributes, followed by "::", to precede dummy argument names in procedure headers, would allow this style. (It would not become required.) If any argument includes such a declaration, they all must (so as to avoid ambiguity between type and attribute names, and argument names).

For example, the following two interface body declarations would be equivalent:

```
SUBROUTINE SUB ( A, B )
  REAL(KIND(0.0E0)), INTENT(IN) :: A
  INTEGER, INTENT(INOUT), OPTIONAL :: B
END SUBROUTINE SUB

SUBROUTINE SUB ( REAL(KIND(0.0E0)), INTENT(IN) :: A, &
                 INTEGER, INTENT(INOUT), OPTIONAL :: B )
END SUBROUTINE B
```