Date:	24 July 1998
To:	J3
From:	Van Snyder
Subject:	Specifications, syntax and edits for SELECT TYPE
References:	98-137, 98-152r1

1 Background

Paper 98-137 illustrated the difficulty of selecting the action to be done depending on the dynamic type of a polymorphic object by using only presently approved mechanisms – one needs to create new non-polymorphic pointer objects, and pointer-assign the polymorphic object to the appropriate non-polymorphic pointer in order access the components.

This paper proposes edits for the improved type-safe mechanism suggested in papers 98-137 and 98-152r1.

2 Specifications

Provide a control construct similar to SELECT CASE to select the type. In addition, declare a variable name that is associated with the expression as if by argument assignment, and that has the appropriate type in each branch of the control construct.

3 Syntax and Semantics

Paper 98-137 proposed most of the following syntax and semantics. Parameterized types were not discussed in 98-137, but are discussed here. Some additional flexibility is proposed.

```
R815a select-type-construct   is select-type-stmt
                                    [ type-guard-stmt
                                      scoped-block ] ...
                                    end-select-stmt
R815b select-type-stmt        is [select-construct-name :] SELECT TYPE ( expr ) ■
                                 ■ ASSOCIATE ( associate-name )
R815c type-guard-stmt         is TYPE IS ( type-spec ) [ select-construct-name ]
                              or TYPE IN ( type-spec ) [ select-construct-name ]
                              or TYPE DEFAULT [ select-construct-name ]
R815d scoped-block            is block
```

During execution of the scoped block, the associate name is associated to the expression as if the associate name were a dummy argument and the expression were an actual argument associated to the associate name by argument association. In addition, the associate name assumes the rank and extents of the expression. If the associate name is assigned, the expression shall be assignable.

The scoped block following a TYPE IS type guard statement is executed if the type parameters and dynamic type of the expression are the same as the type specified in the TYPE IS type guard statement. Within the scoped block the associate name is assumed to have the type and type parameters named in the TYPE IS type guard statement.

The scoped block following a TYPE IN type guard statement is executed if the type and type parameters of the expression are not the same as for a type specified in any TYPE IS type guard statement, the type of the expression is an extension of the type name in the TYPE IN type guard statement and has the same parameters, and no type specified in any other TYPE IN type guard statement is a nearer ancestor of the type of the expression and has the same parameters. Within the scoped block the associate name is assumed to be a polymorphic object of the class named in the TYPE IN type guard statement, and having the same type parameters.

The scoped block following the TYPE DEFAULT type guard statement is executed if no scoped block following a TYPE IS or TYPE IN type guard statement is executed. Within the scoped block the associate name is assumed to have the declared type and type parameters of the expression.

If the expression does not have an extensible type there shall be only one TYPE IS type guard statement, or a TYPE DEFAULT type guard statement. In a TYPE IS type guard statement, the type specified shall be the same type, and have the same type parameters, as the expression.

Note: It is not necessary for the type of the expression to be a derived type.

There is a potential difficulty in recognizing that "TYPE DEFAULT" is a type guard statement. It has exactly the same form as the declaration of a type named DEFAULT. It is not ambituous if one considers that the TYPE DEFAULT type guard statement is only allowed to occur immediately within the scope of a SELECT TYPE construct, while a TYPE DEFAULT type declaration statement would not be allowed at that position. It would be possible to resolve the ambiguity by reversing the keywords, *viz.* IN TYPE, IS TYPE and DEFAULT TYPE, but the last would then not be consistent with CASE DEFAULT.

3.1 Alternative syntax for type guard statements

Remove "ASSOCIATE (*associate-name*)" from the type select statement.

R815c *type-guard-stmt*	**is**	*type-spec* [::] *associate-name*
	or	TYPE DEFAULT [::] *associate-name*

Replace "TYPE IS" by "not CLASS" and "TYPE IN" by "CLASS".

4 Edits

Edits refer to 98-007r2. 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.

[11:47-50]

(2) A procedure interface body (12.3.2.1), excluding any scoping units in it, or

(3) A program unit (11) or subprogram, excluding any scoping units in it, or

(4) A select type construct [new section] (8.1.4), excluding any scoping units in it.

[Editor: Add the following in the same paragraph. Add **ancestor type** to the index.] [47:36+]

An **ancestor type** of an extended type is its parent type, or an ancestor type of its parent type. Among a set of ancestor types, the **nearest ancestor type** is the one that is not an ancestor type of any of the others.

[Editor: Replace "*derived-type-spec*" by "*derived-type-ref*".] [48:36]

[Editor: Replace "*derived-type-spec*" by "*derived-type-ref*".] [49:21]

R502 type-spec	**is**	*intrinsic-type-spec*	[53:18+]
	or	*derived-type-spec*	
R502a *intrinsic-type-spec*	**is**	INTEGER [*kind-selector*]	[53:19]
R502b *derived-type-spec*	**is**	TYPE (*derived-type-ref*)	[53:25]

[Editor: Change *case-construct-name* to *select-construct-name*.] [135:44-45]

Constraint: If a *case-stmt* is identified by a *select-construct-name*, the corresponding *select-case-stmt* shall specify the same *select-construct-name*. [135:45+]

[Editor: Change *case-construct-name* to *select-construct-name*.] [136:1]

Constraint: If the select statement of a select construct is identified by a *select-construct-name*, the corresponding end select statement shall specify the same *select-construct-name*. If the select statement of a select construct is not identified by a *select-construct-name*, the corresponding end select statement shall not specify a *select-construct-name*. [136:2-7]

[Editor: Add a new section.] [138:16+]

8.1.4 SELECT TYPE construct

The SELECT TYPE construct selects for execution at most one of its constituent blocks.

8.1.4.1 Form of the SELECT TYPE construct

```
R815a select-type-construct    is  select-type-stmt
                                     [ type-guard-stmt
                                       scoped-block ] ...
                                     end-select-stmt
R815b select-type-stmt         is  [select-construct-name :] SELECT TYPE ( expr ) ■
                                     ■ ASSOCIATE ( associate-name )
R815c type-guard-stmt          is  TYPE IS ( type-ref ) [ select-construct-name ]
                               or  TYPE IN ( type-ref ) [ select-construct-name ]
                               or  TYPE DEFAULT [ select-construct-name ]
```

Constraint: If a *type-guard-stmt* is identified by a *select-construct-name*, the corresponding *select-type-stmt* shall specify the same *select-construct-name*.

Constraint: For a given select type construct, no two type specifications shall specify the same type and kind type parameters.

Constraint: For a given select type construct, there shall be at most one TYPE DEFAULT type guard statement.

```
R815d type-ref                 is  intrinsic-type-spec
                               or  derived-type-ref
R815e scoped-block             is  block
```

Constraint: If the expression does not have an extensible type, there shall be a TYPE DEFAULT type guard statement, or one TYPE IS type guard statement in which the type specified shall be the same type, and have the same kind type parameters, as the expression.

Note: It is not necessary for the type of the expression to be a derived type.

8.1.4.2 Execution of a SELECT TYPE construct

Execution of the SELECT TYPE statement causes the expression to be evaluated.

A SELECT TYPE construct selects at most one scoped block to be executed. During execution of that block, the *associate-name* is associated (14.6.1.4) to the expression. The *associate-name* assumes the rank and extents of the expression. If the *associate-name* is assigned, the expression shall be assignable.

If the kind type parameters and dynamic type of the expression are the same as the kind type parameters and type specified in a TYPE IS type guard statement, the scoped block following that TYPE IS type guard statement is executed. Within the scoped block the *associate-name* has the type and type parameters specified in the TYPE IS type guard statement.

If the kind type parameters and dynamic type of the expression are not the same as the kind type parameters and type specified in any TYPE IS type guard statement, the dynamic type of the expression is an extension of the type specified in a TYPE IN type guard statement and has the same kind type parameters, and the type specified is a nearer ancestor to the type of the expression than the type with the same kind type parameters specified in any other TYPE IN type guard statement, the scoped block following the TYPE IN type guard statement is executed. Within the scoped block the *associate-name* is a polymorphic object (5.1.1.8) of the class named in the TYPE IN type guard statement, and has the same type parameters.

The scoped block following the TYPE DEFAULT type guard statement is executed if no scoped block following a TYPE IS or TYPE IN type guard statement is executed. Within the scoped block the *associate-name* has the declared type and type parameters of the expression.

Execution of the scoped block, or failure to select a scoped block, completes execution of the construct.

A SELECT TYPE statement shall not be a branch target statement. It is permissible to branch to an END SELECT statement only from within the SELECT TYPE construct.

8.1.4.3 Examples of SELECT TYPE constructs

Note 8.x

```
TYPE, EXTENSIBLE :: POINT
  REAL :: X, Y
END TYPE POINT
TYPE, EXTENDS(POINT) :: POINT_3D
  REAL :: Z
END TYPE POINT_3D
TYPE, EXTENDS(POINT) :: COLOR_POINT
  INTEGER :: COLOR
END TYPE COLOR_POINT

TYPE(POINT), TARGET :: P
TYPE(POINT_3D), TARGET :: P3
TYPE(COLOR_POINT), TARGET :: C
CLASS(POINT), POINTER :: P_OR_C

P_OR_C => C
SELECT TYPE ( P_OR_C ) ASSOCIATE ( A )
TYPE IN ( POINT )
```

```
    ! "CLASS ( POINT ) :: A" assumed here
    PRINT *, A%X, A%Y ! This block gets executed
  TYPE IS ( POINT_3D )
    ! "TYPE ( POINT_3D ) :: A" assumed here
    PRINT *, A%X, A%Y, A%Z
  END SELECT

  P_OR_C => P3
  SELECT TYPE ( P_OR_C ) ASSOCIATE ( A )
  TYPE IN ( POINT )
    ! "CLASS ( POINT ) :: A" assumed here
    PRINT *, A%X, A%Y
  TYPE IS ( POINT_3D )
    ! "TYPE ( POINT_3D ) :: A" assumed here
    PRINT *, A%X, A%Y, A%Z ! This block gets executed
  END SELECT
```

Note 8.y

```
  SELECT TYPE ( EXP(-(X**2+Y**2)) * COS(THETA) ) ASSOCIATE ( Z )
  TYPE IS ( REAL )
    PRINT *, A+Z(1:3), A-Z(1:3)
  END SELECT TYPE
```

[Editor: Add ", associate names [new section] (8.1.4)" after "(14.1.3)".] [303:34]

[Editor: Replace "three" by "four" (and remove the blank before the colon). Remove "and".] [310:13]

[Editor: add ", and select construct association" after first "association".] [310:13]

[312:41+]

14.6.1.4 Select type construct association

Execution of a select type construct establishes an association between the expression and the associated name in the select type statement. The associated name remains associated to the expression throughout execution of the construct. Throughout execution of the select type construct, the expression is known by, and may be accessed by the associated name. Upon termination of execution of the select type construct, the association is terminated.