Date: 10 August 1998
To: J3
From: Van Snyder
Subject: Edits for SELECT TYPE
References: 98-137

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 to access the components.

This paper proposes edits for the improved type-safe mechanism in paper 98-137, approved at meeting 145.

There are extensive differences of detail between this paper and the previous one. In particular, the kind type parameter is not used as part of the selection mechanism – this was nonsense in the previous paper. Section 3.1 – a suggestion to the data subgroup – has been deleted.

2 Specifications

Paper 98-137 described a control construct similar to SELECT CASE to select the type. In addition, a variable name that is associated with the expression as if by argument association, and that has the appropriate type in each branch of the control construct, is declared.

3 Syntax and Semantics

Paper 98-137 described most of the following syntax and semantics. Some additional flexibility is proposed.

R815a *select-type-construct* **is** *select-type-stmt*
[*type-guard-stmt*
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-name*)■
■ [*select-construct-name*]
or TYPE IN (*type–name*) ■
■ [*select-construct-name*]
or TYPE DEFAULT [*select-construct-name*]

The *associate-name* shall not be omitted if the *expr* does not consist of a *name*. If *associate-name* is omitted it is assumed to be the same as the *name* that constitutes the *expr*.

During execution of the 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

type parameters, rank, and extents of the expression. If the associate name is assigned, the expression shall be assignable.

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

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

The block following the TYPE DEFAULT type guard statement is executed if no block following a TYPE IS or TYPE IN type guard statement is executed. Within the block the associate name is assumed to have the declared type 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 named shall be the same as the type of the expression.

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

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.

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

[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*
block] ...
end-select-stmt

R815b *select-type-stmt* **is** [*select-construct-name* :] SELECT TYPE ■
■ (*expr*) [ASSOCIATE (*associate-name*)]

Constraint: The *associate-name* shall be specified if *expr* does not consist only of a *name*.

R815c *type-guard-stmt* **is** TYPE IS (*type-name*) ■
■ [*select-construct-name*]
or TYPE IN (*type-name*) ■
■ [*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, two type guard statements shall not specify the same type name.

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

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 named shall be the same as the type of the expression.

If the *associate-name* is not specified it is assumed to be the same as the *name* that constitutes the *expr*.

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 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 type parameters. rank, and extents of the expression. If the *associate-name* is assigned, the expression shall be assignable.

If the dynamic type of the expression is the same as the type named in a TYPE IS type guard statement, the block following that TYPE IS type guard statement is executed. Within the block the *associate-name* has the type named in the TYPE IS type guard statement.

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

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

Execution of the block, or failure to select a 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
```

The following example illustrates the *associate-name* assumed to be the same as the *name* that constitutes the *expr*. Note 8.y

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

The following example illustrates the use of a `SELECT TYPE` construct with the type of the expression not a derived type. Note 8.z

```
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: Add a new paragraph.] [309:40+]

The name that appears as an associate name in a SELECT TYPE construct has a scope of the construct. It is a variable having the type, type parameters, rank and extents specified in 8.1.4.2.

[Editor:Replace the first sentence by the following:] [310:13-15]

There are four forms of name association: argument association, use association, host association, and construct association.

[312:41+]

14.6.1.4 Construct association

Execution of a select type construct establishes an association between the expression and the associate name in the select type statement. The associate 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 associate name. Upon termination of execution of the select type construct, the association is terminated.