TECHNICAL CORRIGENDUM 5

Page and line numbers on the right refer to J3/04-007. The interpretation upon which the edit is based is shown in italics. Page and line numbers are not relevant for ISO/IEC 1539-1:2004 or for WG5–N1601. For these documents subclause and paragraph should be used to locate the edit.

[41:11-12] *f95/0098*

Subclause 4.4.4.1
In constraint C417, change "unless ... dummy function" to "unless it is of type CHARACTER and is the name of a dummy function or the name of the result of an external function".

[41:34,36] *f95/0098*
At the end of subclause 4.4.4.1 in list item (4), after "invoking the function" insert "or passing it as an actual argument"; change "host or use" to "argument, host, or use".

[47:11] *f03/0122*

Subclause 4.5.1.3
In the last sentence of the second paragraph, delete "declared to be PRIVATE or".

[67:19+] *f03/0090*

Subclause 4.7
Following constraint C493, add a new constraint:

C493a (R469) An *ac-value* shall not be unlimited polymorphic.

[67:21] *f03/0090*
In constraint C494, replace "same type" by "same declared type".

[68:9] *f03/0090*
In the second paragraph of the subclause, after "; in this case, the", replace "type and type parameters" by "declared type and type parameters".

[68:11] *f03/0090*
In the first sentence of the third paragraph of the subclause, replace "type and type parameters" by "declared type and type parameters".

[68:14+] *f03/0090*
Following the third paragraph of the subclause, insert the following new paragraph:

The dynamic type of an array constructor is the same as its declared type.

[81:15] *f03/0127*

Subclause 5.1.2.7
In the first sentence of the second paragraph, change "during the execution' to "during the invocation and execution".

[81:26] *f03/0127*
Append the following sentence to the third paragraph: "Any undefinition or definition implied by association of an actual argument with an INTENT(OUT) dummy argument shall not affect any other entity within the statement that invokes the procedure."

[93:9] *f03/0134*

Subclause 5.3
In the first sentence of the fourth paragraph, after "intrinsic function," insert "is not a component,".

[97:7+] *f03/0131*

Subclause 5.5.1.1
Insert the following new paragraph at the end of the subclause:

> If any data object in an *equivalence-set* has the SAVE attribute, all other objects in the *equivalence-set* have the SAVE attribute; this may be confirmed by explicit specification.

[98:17] *f03/0063*

Subclause 5.5.2
In rule R558, delete the second line: **or** *proc-pointer-name*

[98:18] *f03/0063*

In constraint C587, delete "or *proc-pointer-name*"

[98:21] *f03/0063*
[98:22] *f03/0133*

In constraint C588, after "that is allocatable," insert "a procedure pointer," and after "BIND attribute," insert "an unlimited polymorphic pointer,".

[98:25] *f03/0063*

In constraint C590, delete "or *proc-pointer-name*".

[100:12-15] *f03/0063*

Subclause 5.5.2.3
In the sixth paragraph of the subclause, delete the third, fourth and fifth sentences: "A procedure pointer shall be storage ... type parameters.".

[116:25] *f03/0024*

Subclause 6.3.3.2
After the first sentence of the second paragraph, insert the new sentence: "The pointer shall have the same dynamic type and type parameters as the allocated object, and if the allocated object is an array the pointer shall be an array whose elements are the same as those of the allocated object in array element order."

[144:5-6] *f03/0138*

Subclause 7.4.2
In constraint C727, change "an external, module," to "a module" and change "or a procedure pointer" to "a procedure pointer, or an external procedure that is accessed by use or host association and is referenced in the scoping unit as a procedure, or that has the EXTERNAL attribute".

[193:13-15] *f03/0132*

Subclause 9.5.2
Replace the twelfth paragraph of the subclause ("If a derived-type list item … input/output procedure.") by the following new paragraph:

> If a derived-type list item is not processed by a user-defined derived-type input/output procedure and is not treated as a list of its individual components, all of the subcomponents of that list item shall be accessible in the scoping unit containing the input/output statement and shall not be pointers or allocatable.

[260:2] *f03/0141*

Subclause 12.3.2.1
In the last sentence of the paragraph beginning "If an explicit specific interface ...", i.e. the sixth paragraph of the subclause, after "scoping unit" insert ", except that if the interface is accessed by use association, there may be more than one local name for the procedure". Append

a sentence at the end of the paragraph: “If a procedure is accessed by use association, each access shall be to the same procedure declaration or definition.”.

[261:3] *f03/0071*

At the end of the paragraph beginning “A generic interface block specifies ...”. i.e. the ninth paragraph of the subclause, add the following sentence: “If a specific procedure in a generic interface has a function dummy argument, that argument shall have its type and type parameters explicitly declared in the specific interface.”.

[263:13+16+] *f03/0112*

Subclause 12.3.2.1.2

At the end of the subclause, after Note 12.10, add the following new note:

> NOTE 12.10a
> If the second argument of a procedure specified in a defined assignment interface block has the POINTER or ALLOCATABLE attribute, it cannot be accessed by defined assignment, since the right-hand side of the assignment is enclosed in parentheses before being associated as an actual argument with the second argument. This makes it an expression, which does not have the POINTER or ALLOCATABLE attribute.

[271:28] *f03/0137*

Subclause 12.4.1.3

Append the following two sentences at the end of the fifth paragraph of the subclause, that is after “function procedure pointer, or dummy procedure.” : “If both the actual argument and dummy argument are known to be functions, they shall have the same type and type parameters. If only the dummy argument is known to be a function, the function that would be invoked by a reference to the dummy argument shall have the same type and type parameters, except that an external function with assumed character length may be associated with a dummy argument with explicit character length.”.

[275:2,5] *f03/0127*

Subclause 12.4.1.7

In list item (2) of the first paragraph, in each of the first and second sentences change “during the execution” to “during the invocation and execution”.

[276:36+] *f03/0135*

Subclause 12.4.4

Add the following new list item after item (2) (a):

> (a2) if that scoping unit is of a subprogram that defines a procedure with that name;

[278:15+] *f03/0135*

Subclause 12.4.4.2

Add the following new list item after item (3):

> (3a) If the scoping unit is of a subprogram that defines a procedure with that name, the reference is to that procedure.

[286:22+] *f03/0126*

Subclause 12.6

Following constraint C1271, add the following new constraint:

> C1271a The *designator* of a variable with the VOLATILE attribute shall not appear in a pure subprogram.

[287:17, 288:1] *f03/0119*

Subclause 12.7.1
In constraint C1278, after "scalar" change "and" to a comma and at the end of the sentence insert, ", and shall not have a type parameter that is defined by an expression that is not an initialization expression".

[316:16-17] *f03/0125*

Subclause 13.7.38
In the **Arguments** paragraph, in each of the two argument descriptions, replace "of extensible type" by "of extensible declared type or unlimited polymorphic".

[316:21-22] *f03/0125*

In the **Result Value** paragraph, after the second "otherwise" insert "if the dynamic type of A or MOLD is extensible," and at the end of the sentence insert "; otherwise the result is processor dependent".

[347:30, 348:1] *f03/0125*

Subclause 13.7.101
In the **Arguments** paragraph, in each of the two argument descriptions, replace "of extensible type" by "of extensible declared type or unlimited polymorphic".

[348:3,4] *f03/0125*

In the **Result Value** paragraph, change "The result" to "If the dynamic type of A or B is extensible, the result" and append the following new sentence to the paragraph: "If neither A nor B has extensible dynamic type, the result is processor dependent."

[363:9-10] *f03/0022*

Clause 14
In the first sentence of the second paragraph, replace "for all kinds of real and complex data" with "for all kinds of real and complex IEEE floating-point data".

[376:17+] *f03/0034*

Subclause 14.10.12
At the end of the **Result Value** paragraph, add the following two new cases:

Case (iii): If IEEE_SUPPORT_INF(X) is true and X is infinite, the result is +infinity.

Case (iv): If IEEE_SUPPORT_NAN(X) is true and X is a NaN, the result is a NaN.

[395:8] *f03/0129*

Subclause 15.1.2.5
In list item (1) of the **Argument** paragraph, before "type parameters" insert "kind".

[395:16+] *f03/0129*

At the end of the **Argument** paragraph, insert the following new sentence after the list: "X shall not be a zero-length string.".

[396:5-7] *f03/0129*

Subclause 15.2.1
In the second sentence of the first paragraph of the subclause replace "; if the type is character ... one." by ". If the type is character, the length type parameter is interoperable if and only if its value is one.".

[399:2-3] *f03/0129*

Subclause 15.2.4
Replace the first paragraph by the following paragraph:

> A named scalar Fortran variable is **interoperable** if and only if its type and type parameters are interoperable, it has neither the pointer nor the allocatable attribute, and if it is of type character its length is not assumed or declared by an expression that is not an initialization expression.

[399:7-8] *f03/0129*

Subclause 15.2.5
Replace the first paragraph by the following paragraph:

> A Fortran variable that is a named array is **interoperable** if and only if its type and type parameters are interoperable, it is of explicit shape or assumed size, and if it is of type character its length is not assumed or declared by an expression that is not an initialization expression.

[407:28] *f03/0136*

Subclause 16.2.3
In the second paragraph, after "**distinguishable** if" insert "one is a subroutine and the other is an array, or if".

[409:19] *f03/0140*

Subclause 16.3
In the second sentence of the second paragraph, before "scoping unit that includes" insert "innermost executable construct or".

[409:26] *f03/0140*

In the second sentence of the third paragraph, before "scoping unit that includes" insert "innermost executable construct or".

[411:21] *f03/0063*

Subclause 16.4.1.3
Delete list item (7): A *proc-pointer-name* in a *common-block-object* in a *common-stmt*;

[416:23] *f03/0063*

Subclause 16.4.3.1
In list item (8) of the second paragraph, replace "A pointer" by "A data pointer".