J3 Responses to Interpretation Requests from Japan

J3 has looked at the interpretation requests of Japan in J3 paper 99-208. We have put these requests into the following classes:

- "Fortran 95 Interpretation" - JOR subgroup will submit this item as a formal interpretation.
- "Recommended for Fortran 2000" - These items are problems, but they are not serious enough to warrant a formal Fortran 95 Interpretation.
- "Deferred to Editor" - We will recommend that the editor of Fortran 2000 include these changes into the text of the Fortran 2000 draft.
- "Not Recommended" - We believe that no action is required on these items. An explanation is provided.

. Here are the responses of J3, which are indicated by a ">>" symbol at the beginning of the line.

All references are to Fortran 95.

J3 thanks the ITSCJ for their careful reading of Fortran 95. Please feel free to contact J3 if you disagree with any of these recommendations.

JP-1)
In 2.5.7 Intrinsic, the third sentence:

"All intrinsic data types, procedures, and operators may be used in any scoping unit without further definition or specification."

Insert 'assignments' after 'procedures,'.

>> Recommended for Fortran 2000.

JP-2)
In the sentence immediately after NOTE 4.18:

"If initialization-expr appears for a nonpointer component, that component in any object of the type is initially defined or becomes defined as specified in (14.7.5) with the value determined from initialization-expr."

Change '(14.7.5)' to '14.7.5'.

>> Recommended for Fortran 2000.

JP-3)
4.4.4 Construction of derived-type values
Before NOTE 4.33:

"A structure constructor whose component values are all constant expression is a derived-type constant expression."

In the above, 'derived-type constant expression' should be bold, because the term is defined here.

>> Deferred to editor.

JP-4)
4.4.4 Construction of derived-type values
Before NOTE 4.34:
"Where a component in the derived type is a pointer, the corresponding constructor expression shall evaluate to an object that would be an allowable target for such a pointer in a pointer assignment statement(7.5.2)."

Change 'an object' to 'a result value'. A value of an expression can not be an object, by definition.

>> Not Recommended. We believe "object" is correct. See the constraints after R737 in section 7.5.2.

JP-5)
4.5 Construction of array values,

"The ac-do-variable of an ac-implied-do that is in another ac-implied-do shall not appear as the ac-do-variable of the containing ac-implied-do."

This sentence should be a Constraint.

>> Fortran 95 Interpretation.

JP-6)
5.1 Type declaration statements

As for the 16th and 19th constraint after R506:

the 16th:
"Constraint: The function-name shall be the name of an external function, an intrinsic function, a function dummy procedure, or a statement function."

Because the syntactic class 'object-name' is only defined as a 'name' in the standard, the following constraint should be added here:
Constraint: The object-name shall be the name of a data object.

After that, in the 19th:
"Constraint: initialization shall not appear if object-name is dummy argument, a dummy argument, a function result, an object in a named common block unless the type declaration is in a block data program unit, an object in blank common, an allocatable array, an external name, an intrinsic name, or an automatic object."

In the above, 'a function result,' should be removed.

If we can not add the constraint above, 'a statement function' should be added in the 19th constraint.

>> Fortran 95 Interpretation.

JP-7) is absent

JP-8)
5.1 Type declaration statements

After NOTE 5.3:

"If a length-selector (5.1.1.5) is a nonconstant expression, ..."

Change 'length-selector' to 'char-selector', 'char-len-selector' or 'character-length'.

>> Fortran 95 Interpretation.

JP-9)
5.1.1.5 4th Constraint after R510 (Page 51 Line 1) states that:

"Constraint: A function name declared with an asterisk
char-len-param-value shall not be array-valued,
recursive, or pure."

The word "Constraint" should be shown in distinguishing type font because this is a constraint for an obsolescent feature.

>> Recommended for Fortran 2000.

JP-10)
5.1.1.5 After the NOTE 5.6 (Page 51 Line 32,33) states that:

"The length specified for a character-valued statement function or statement function dummy argument of type character shall be a constant specification expression."

This should be shown as a constraint because it is a restriction for character length.

Note that this is an obsolescent feature.

>> Recommended for Fortran 2000.

JP-11)
5.1.2.4.1 After R516 (Page 54 Line 29-33) states that:

"Constraint: An explicit-shape array whose bounds depend on the
values of nonconstant expressions shall be a dummy
argument, a function result, or an automatic array
of a procedure.

An automatic array is an explicit-shape array that is declared in a subprogram, is not a dummy argument, and has bounds that are nonconstant specification expressions."

The constraint seems meaningless because the following paragraph
which defines the automatic array duplicates it.
Is this constraint necessary ?

>> Not Recommended. Yes, we believe that the constraint is necessary because it belongs to three things, not just automatic arrays. No change is necessary.

JP-12)
5.1.2.4.3 (2) after R518 (Page 55 Line 41) states that:

"(2) They are specified in a pointer assignment statement. ..."

In this description, the term "pointer assignment statement" should
be changed to "pointer assignment".

Reason : The bounds of each dimension of an array pointer may be
specified not only in a pointer assignment statement but also in a
derived-type intrinsic assignment statement with a component of an
array pointer.

>> Fortran 95 Interpretation.

JP-13)
5.2.10 5th and 9th constraints after R537 (Page 62 Line 2 and 10)
state that:

"Constraint: A scalar-int-expr of a data-implied-do shall involve
as primaries only constants, ... , or
DO variables of the containing data-implied-dos, ..."

"Constraint: In an array-element or a scalar-structure-component
that is a data-i-do-object, any subscript shall be an
... ,or DO variables of the containing
data-implied-dos, ..."

In the latter constraint, the phrase :
"DO variables of the containing data-implied-dos"
should be changed to :
"DO variables of this data-implied-do and the containing
data-implied-dos".

Consider the following program:

```
INTEGER, DIMENSION (3,3) :: IARY
DATA ((IARY(IA,JA), JA=IA,3), IA=1,3) /1,2,3,4,5,6/
```

The "IA" in "JA=IA,3" is a scalar-int-expr described in the former
constraint stated above. In this case, "IA" is a DO variable of the
containing data-implied-do.
This is the meaning of the phrase "containing data-implied-do".
In another word, "containing" should mean "outer".

The "IA" and "JA" in "IARY(IA,JA)" are subscripts described in the latter constraint stated above. In this case, "IA" is a DO variable of the containing data-implied-do. However, "JA" is a DO variable of not "containing" but "this" data-implied-do.

If the "containing data-implied-do" means both "this" and "outer" data-implied-do, the former constraint is incorrect because it allows the DO variable of this data-implied-do (JA of "JA=IA,3" in this example) to be involved in scalar-int-expr (IA and 3 of "JA=IA,3" in this example).

>> Recommended for Fortran 2000.

JP-14)
5.2.10 R539 (Page 62 Line 13,14) states that:

"R539 data-stmt-repeat is scalar-int-constant
or scalar-int-constant-subobject"

The syntactic definition of scalar-int-constant-subobject can be derived from int-constant-subobject (1.6.3).
Add the definition of int-constant-subobject.

>> A *constant-subobject* is a subobject of a named constant. We will add clarifying text to Fortran 2000.

JP-15)
5.2.10 3rd paragraph after R540 and constraints (Page 62 Line 43-45) states that:

"A zero-sized array or an implied-DO list with an iteration count of zero contributes no variables to the expanded sequence of variables, but a zero-length scalar character variable does contribute a variable to the list."

The word "list" at the end of above statement should be replaced with "sequence" or "expanded sequence".

Note that the words "list" and "sequence" are used for the separate meanings in this section as follows :

"The data-stmt-object-list is expanded to form a sequence of pointers and ..."
"The data-stmt-value-list is expanded to form a sequence of data-stmt-constants."

Also refer to 2.5.9 for the definition of "sequence".

>> Recommended for Fortran 2000.

JP-16)
5.4 2nd paragraph after R545 and constraints (Page 66 Line 11)

states that:

"Any namelist-group-name may occur in more than one NAMELIST statement in a scoping unit."

Can a namelist-group-name occur more than once in one NAMELIST statement ?

Is the following NAMELIST statement standard conforming ?

```
NAMELIST /NLIST/ A, B   /NLIST/ C, D
```

If this is standard conforming, is it the same as the following ?

```
NAMELIST /NLIST/ A, B, C, D
```

>> Fortran 95 Interpretation.

JP-17)
5.4 3rd paragraph after R545 and constraints states that:

"A namelist group object may be a member of more than one namelist group."

Can a namelist group object occur more than once in one namelist group?

Is the following NAMELIST statement standard conforming?

```
NAMELIST /NLIST/A,B,A
```

>> Fortran 95 Interpretation.

JP-18)
In 7.1.1.1, a constraint below R702 states that:

"subobject shall be a subobject designator whose parent is a constant. A variable that is a primary shall not be a whole assumed-size array."

The second sentence,'A variable that ...',should be 'Constraint: A variable that...'.

>> Recommended for Fortran 2000.

JP-19)
The third paragraph of 7.1.4.1 has the following:

"If a pointer appears as one of the following, the associated target object is referenced:
(1) A primary in an intrinsic or defined operation,
(2) As the expr of a parenthesized primary, or

(3) As the only primary on the right-hand side of an intrinsic assignment statement."

The first word "As" in (2) and (3) should be removed.

>> Deferred to editor.

JP-20)
In 7.5.3.2, the paragraph before NOTE 7.49 states that

"A statement that is part of a where-body-construct shall not be a branch target statement."

The term 'branch target statement' is defined to be one of a specific set of statements, and is not appropriate here. Replace this with a constraint such as:

"Constraint: A statement label of any statement that is part of a where-body-construct shall not be referred to from outside of the construct."

The same applies to 7.5.4.1 (FORALL).

>> Recommended for Fortran 2000.

JP-21)
NOTE 7.49 has the following example:

```
WHERE (A > 0.0)
      A = LOG (A)
      A = A / SUM (LOG (A))

END WHERE
```

But SUM(LOG(A)) causes an error when the array variable A has negative values. So the example program should be changed to something like the following.

```
WHERE (D > 0.0)
      X = -B + SQRT (D)
      R = D / SUM (SQRT (ABS (D)))

END WHERE
```

>> Not Recommended. The purpose of the example is to show that LOG(A) will be referenced even if that was not the intent of the programmer.

JP-22) is absent

JP-23)
In NOTE 7.55, the "END FORALL" statement should be supplied.

>> Not Recommended. The purpose of the example was to illustrate the FORALL mask and the ellipses indicate the continuation of the rest of the block.

JP-24)
In 8.1.4.1.2, second constraint below R833 states that:

"The do-term-shared-stmt shall be identified with a label
and all of the label-do-stmts of the shared-term-do-construct
shall refer to the same label."

This implies a label-do-stmts of the outer-most outer-shared-do-construct will permit not to refer to the same label, because shared-term-do-construct does not include outer-most outer-shared-do-construct.

So the term "shared-term-do-construct" should be changed to "inner-share-do-construct and outer-shared-do-construct."

>> Fortran 95 Interpretation.

JP-25)
In the second sentence of 8.1.4.3:

"Once active, the DO construct becomes inactive only when the
construct it specifies is terminated(8.1.4.4.4)."

Remove "it specifies".

>> Deferred to editor.

JP-26) is absent

JP-27)
In the second sentence from the bottom of 8.2:

"It is permissible to branch to an end-do-stmt or a do-term-action-
stmt only from within its DO construct."

"end-do-stmt" should be "end-do".

>> Recommended for Fortran 2000.

JP-28)
The last sentence of 8.2.1:

"Only branch target statements (8.2), FORMAT statements, and DO
terminations shall be referred to by the use of statement labels
(3.2.4)."

"(3.2.4)" should be moved to the end of the previous sentence.

>> Recommended for Fortran 2000.

JP-29)
The last sentence of 8.2.1:

"Only branch target statements (8.2), FORMAT statements, and DO terminations shall be referred to by the use of statement labels (3.2.4)."

Change ", FORMAT statements, and DO terminations" to "and FORMAT statements". The reasons are as follows:

1. All cases are covered without "DO terminations".
2. A DO termination can be a shared-term-do-construct. A statement label is intended to refer to a statement, and the notion of referring to a construct by a statement label is not defined.

>> Recommended for Fortran 2000.

JP-30)
The first sentence of 9.4.1.7:

"If an end-of-record condition (9.4.3) occurs and no error condition (9.4.3) occurs during execution of an input/output statement that contains an EOR= specifier ..."

"input/output statement" should be "input statement".

>> Recommended for Fortran 2000.

JP-31)
The fourth sentence of 10.8 and sixth sentence of 10.9:

"Each value is either a null value or one of the forms:

```
c
r*c
r*
```

where c is a literal constant or a nondelimited character constant and r is an unsigned, nonzero, integer literal constant."

"a literal constant" should be "an optionally signed literal constant"

>> Fortran 95 Interpretation. Add the phrase, "if integer or real" to that last phrase above.

JP-32)
The first sentence of NOTE 10.25:

"List-directed input/output allows data editing according to the type of the list item instead of by a format specifier."

"format specifier" should be "format specification".

>> Recommended for Fortran 2000.

JP-33)
Page 176/ line 16 (2nd paragraph before 10.8.1.1):

"(3) The first nonblank character"
should be
"(3) The first character".

REASON:
Since (1) says that "the character sequence does not contain the value separators blank, comma, or slash," the first character should not be a blank. So the word "nonblank" is confusing.

>> Recommended for Fortran 2000.

JP-34)
Page 177/ line 35 (6th line of 10.8.2):

The word "sequence" in "not occur within a constant or sequence"
should be "character sequence".

>> Recommended for Fortran 2000. Rewrite [177:34-36] to read, "The processor may begin new records as necessary, but the end of record shall not occur within a constant except for complex constants and character sequences. The processor shall not insert blanks within a constant or character sequence." Check NAMELIST output in section 10.9.2.

JP-35) is absent

JP-36)
Page 192/ line 17 (The last line of 12.2.1.1):

In dummy data objects, the size is allowed to be assumed.
However the characters of "size," is still small.
It should be changed to normal size.

>> Deferred to editor.

JP-37)
Page 196/ line 3 (The first statement of 12.3.2.1.1):

The referring section "(12.4)" is wrong, and should be
"(7.1.3, 7.3)".

>> Recommended for Fortran 2000.

JP-38) is absent

JP-39)
Page 204/ line 17 (NOTE 12.20):

In 6.3.3.2, "If a pointer is currently associated with an
allocatable array, the pointer shall not be deallocated".
So "DEALLOCATE (B)" would NOT be permitted.

>> Fortran 95 Interpretation.

JP-40) is absent

JP-41)
p.278, in the item (3) of 14.1.2.4:

(3) A procedure is not established in a scoping unit if it is neither established to be generic nor established to be specific.

Change "procedure" to "procedure name".

>> Deferred to editor.

JP-42)
p.289, in the item (6) of 14.7.5:

(6) A reference to a procedure causes the entire dummy argument data object to become defined if the entire corresponding actual argument is defined <obsolescent>with a value that is not a statement label.</obsolescent>

Delete the obsolescent-font part.

>> Recommended for Fortran 2000.

JP-43)
p.304, in the following part of B.1.1:

"R901 io-implied-do-control is do-variable = scalar-numeric-expr , ..."

Change "R901" to "R918".

>> Deferred to editor.

JP-44)
p.307, in the following part of B.2.6:

... keyboards with screen displays, it is an unnecessary overhead, and is potentially error-prone, to have to locate positions 6, 7, or 72 on a line.

Change "72" to "73".

>> Deferred to editor.

JP-45)
p.311, in the following part of C.1.2:

... an exponent range from 10**-50 to 10**50.

Change "exponent range" to "range" or "value range".

>> Deferred to editor.

JP-46)
p.311, in the following line of the program in C.1.3:

```
  CURRENT => CURRENT % NEXT_CELL
```

Change "CELL" to "NODE".

>> Deferred to editor.

JP-47)
p.317, in the following line of the program in C.4.6:

```
  X(2:N-1) = (X(1:N-2) + 2*X(2:N-1) + X(3:N+1) ) / 4
```

Change "X(3:N+1)" to "X(3:N)".

>> Deferred to editor.

JP-48)
p.342, in the following sentence in C.11.2.3.2:

```
  This decision will be recorded as the true elements of an array FLIP.
```

Change "FLIP" to "FLIPS".

>> Deferred to editor.