Voting Summary on JTC 1/SC 22 N 3661

P-Member	Approve	Approve w/Comments	Disapprove	Abstain
Austria				
Belgium				
Brazil				
Canada		x		
China				
Czech Republic	x			
Denmark	x			
Egypt				
Finland				
France				x
Germany	x			
Ireland				
Italy	x			
Japan	x			
DPR of Korea				
Republic of Korea	x			
Netherlands	x			
Norway				
Romania				
Russian Federation				
Slovenia				
Switzerland				x
Ukraine				
United Kingdom		x		
United States		x		

Comments:

Canada

Technical Comments:

1.) When extending a type, a non pointer component (the parent component) is created. Its type is that given by the <parent-type-name> in the derived type

declaration. As such, the rules for the <parent-type-name> should follow the rules for declaration of a non pointer component as closely as possible.

In the FCD, the <parent-type-name> must be "the name of an accessible extensible type." This includes any extensible type defined later in the same specification part, or (depending on one's interpretation) the type being defined by the same derived type definition. To follow the rules for non pointer component declaration more closely, the <parent-type-name> should be the name of a previously defined extensible type (see C438).

Proposed edit:
[45:17] Change "accessible" to "previously defined."

2.) Unlimited polymorphic derived type components appear to be prohibited by C438 and C439. This is almost certainly unintended.

Proposed edits:
[50:19] Before "shall" insert "shall be CLASS(*) or"
[50:21] Before "shall" insert "shall be CLASS(*) or"

3.) An optional KIND argument has been added to many intrinsic functions in Fortran 2003. This argument, if present, specifies the kind of parameter of the function result. These functions should not be used as actual arguments because it would be impossible to give the dummy argument the correct interface.

However, the optional KIND argument has been added to both the INDEX and LEN intrinsic functions, which can be used as actual arguments. The new KIND argument should be removed from these intrinsic functions.

Proposed edits:
[322:24] Remove "KIND"
[323:3-4] Change "Integer…default integer type" to "default integer scalar"

Editorial Comments:

1.) There is an error in the example given in section C.1.4. Section 4.5.6.2 (Type-bound procedure overriding) states the following about the overriding binding and the overriding binding: Passed-object dummy arguments, if any, shall correspond by name and position.

The passed-object dummy arguments for the RENDER binding in this example do not correspond by name.

Proposed edits:
[444:7] Change "PASS(TRIANGLE) to "PASS(OBJECT)"
[444:12] Change "(TRIANGLE) to "(OBJECT)"

[444:13] Cange “::TRIANGLE” to “::OBJECT”

United Kingdome

1. Minor technical changes

C1. Array constructor restriction

[68:8] After "," insert "each <ac-value> expression in the array constructor shall have the same length type parameters;"

Rationale: The old F95 restriction on array constructors that each ac-value had to have the same character length was taken out (presumably when the type-spec syntax was added). This would seem to be an error, since now ['a','abc'] is a sort of Schrodinger's array constructor with both length 1 and 3. The F95 restriction is still required when there is no type-spec, but needs updating to cope with PDTs. It is not necessary to say "type and type parameters" because type (and incidentally, kind type parameters but not length ones) are covered by constraint C494 at [67:20-21].

C2. ASYNCHRONOUS and VOLATILE

[77:15] After "scoping units" insert "(11.2.1, 16.4.1.3)".
Rationale: Clarification.

[85:8] After "scoping units" insert "(11.2.1, 16.4.1.3)".
Rationale: Clarification.

[251:12] After "the module" insert "except that an entity may have the ASYNCHRONOUS or VOLATILE attributes in the local scoping unit even if the associated module entity does not".
Rationale: The sentence was incorrect because it ignored the possibility of attributes changing because of asynchronicity or volatility.

[411:5]. After "host" add "except that the local entity may have the ASYNCHRONOUS or VOLATILE attributes even if the host entity does not".
Rationale: The sentence was incorrect because it ignored the possibility of attributes changing because of asynchronicity or volatility.

C3. VOLATILE

[201:11-12]. Delete paragraph.
Rationale: This certainly should not be the case for WRITE since volatile variables are not permitted to be intent(in), see [73:17-18].
For READ, it is disallowed by [199:24-25] and [200:13-14] since volatility is a characteristic, see [256:28].

[271:12-] Add "NOTE 12.26a An object with the VOLATILE attribute may become associated with a dummy argument that does not have the VOLATILE attribute. In this case, the programmer needs to be sure that the actual value does not change by means not specified by the program during the execution of the procedure."

[411:35+3] Replace "If" by "In a module or internal procedure, if"
[411:35+3-4] Delete "other than by an IMPORT statement".

[411:35+5-6] Replace "in the scope ... procedure" by "in the local scope".
Rationale: Clarification.

[421:42]. Change "listed in 5.1.2.16" to "not specified by the program (see 5.1.2.16)".
Rationale: There is no list in 5.1.2.16.

C4. IMPORT

[260:11] Append
"If an entity that is made accessible by this means is accessed by host association and is defined in the host scoping unit, it shall be explicitly declared prior to the interface body."
Rationale: In the case of an IMPORT statement with no list, it is not clear from [260:3-11] exactly which entities must be declared before the interface block.

C5. Interoperability

[279:32] Replace "an" by "a nonoptional".
Rationale: Clarification. The scope of "nonoptional" on this line is ambiguous. Inserting an additional "nonoptional" clarifies matters.

[279:34] Add "scalar" after "interoperable".
Rationale: An interoperable function must have a scalar result (see 400:7); this should be part of the constraint.

[398:8] Add sentence to C1505: "It shall not have the PRIVATE attribute."
Rationale: This is stated at 47:8 and should be part of the constraint.
If it were permitted for a derived type with the BIND attribute to have private components, then by 47:9-11, if identical definitions were placed in separate scopes, they would be different types but both would be interoperable with the same C type. Further, privacy would be lost by accessing the private parts in C.

C6. 'Linkage' of procedures and variables
Rationale: There are several places where "linked" is used for a relationship between a C function and a Fortran procedure with a reference to (12.5.3). The word does not appear in (12.5.3). There is also exactly one place where "linked" is used of variables to mean "associated" (by linkage association). We believe that this term is a relic from an early draft and needs to be removed.

[21:10+5]. Change "linked ... procedure" to "invoked".

[77&78: Note 5.9] In line 5, change "linked ... entity" to "referenced from Fortran". Delete the second paragraph.

[391:6] Change "linked" to "associated".

[431:34-35]. Delete entry for "linked".

C7. VALUE attribute

[73:22-24] Delete constraint C528.
Rationale: This is an unnecessary restriction in the general case: it is needed only for interoperability and the rules for interoperability of types cover that situation. Also, it is inconsistent to restrict character in this way but not to do so for a derived type with a length type parameter.

C8. Binding labels

Rationale:
1. If the BIND(C) attribute is given to a module procedure, it has a binding label even if it is PRIVATE. This allows it to be accessed directly from C and indirectly from Fortran. This loophole needs to be closed.
2. 15.4.1 gives a procedure binding label to a procedure pointer; this would appear to be an oversight.
3. It is proposed that the binding label must be valid as an identifier on the companion processor.

[77:21]. Add "If its value after discarding leading and trailing blanks has nonzero length, it shall be valid as an identifier on the companion processor."

[265:12]. After "binding label" add "or its absence,".

[279:28] Change "or a" to ", a module procedure that does not have the PUBLIC attribute, or a".

[403:4-6] Replace "If a variable ... ignored." by "If a variable or common block has the BIND attribute with the NAME= specifier and the value of its expression after discarding leading and trailing blanks has nonzero length, the procedure has this as its binding label. The case of letters in the binding label is significant."

[403:8] Add "Otherwise the procedure has no binding label." at the end of the paragraph.

[403:32-34] Replace "If a procedure ... ignored." by "If a procedure has the BIND attribute with the NAME= specifier and the value of its expression after discarding

leading and trailing blanks has nonzero length, the procedure has this as its binding label. The case of letters in the binding label is significant."

[403:35] Change "not a dummy procedure" to "not a dummy procedure, not a procedure pointer, and not a module procedure with the PRIVATE attribute".

[403:36] Add "Otherwise the procedure has no binding label." at the end of the paragraph.

C9. Comma before BIND(C)

Rationale: The syntax for the FUNCTION and SUBROUTINE statements (R1224, R1229, and R1232) specifies that there be no comma between ")"
and "BIND", which is consistent with there being no comma ahead of "RESULT" in Fortran 95. However, all the examples have a comma in this place. They should be corrected:

[281:NOTE12.39+5], [393:10+5], [400:20+4], [401:-11], [404:0+7],
[486:10], and [487:32] Delete comma before "BIND".

[404:0+4] Replace comma by "()".

[485:2], [488:35], and [489:16] Delete comma before "&".

2. Editorial

E1. ISO_FORTRAN_ENV module [360:33+] Add new note: "The constants relating to input/output units may take any positive, zero or negative value other than -1. The value -1 is reserved for use as a special indicator in the INQUIRE statement (9.9.1.17)."

Rationale: Clarification. The intention is that INPUT_UNIT etc may take any integer value. It is not clear from the text that it may take any negative value, nor is it obvious at this point why -1 is prohibited.

E2. ISO_C_BINDING module
[391:31] Change "C_COMPLEX" to "C_FLOAT_COMPLEX"
Rationale: This name must be the same as that used in Table 15.2.

E3. Miscellanea

E3.1 Terminology
[3:16-17] Change "logical units" to "external files".
Rationale: The term "logical unit" does not appear in earlier Fortran standards.

E3.2 Typographical errors
[346:20] Change "parameters" to "parameter".

[429:18&20] Add "an" before "explicit".

E3.3 Coding error in C.10
[486:35] "float* f, c" should be "float *c, *f"
Rationale: The first form declares c to be a float, not a pointer to a float. The "*" pointer indicator attaches to variable names, not to the type! Also the names are reversed relative to the Fortran code.

E3.4 Coding error in C.10
[486:38] "pass *arrays" should be "struct pass *arrays"
Rationale: Missing keyword.

E4. Glossary

E4.1 Entries to be updated
Rationale: Entries in this section have not been updated to accommodate new concepts or new syntax in this revision and so are currently incorrect.

E4.1.1 generic identifier
[430:16] Insert before ".":
"or that appears in a GENERIC statement and is associated with the specific type-bound procedures".

E4.1.2 invoke
[431:24+] Insert new case:
"(3) To call a final subroutine by finalization."

E4.1.3 kind type parameter
[431:27] Insert "4.5.2" before ")".
[431:28] Insert before ".":
", or a derived type parameter that is declared to be KIND"

E4.1.4 label
[431:29] After "See" add "binding label or".

E4.1.5 name association
[432:13] Change "or" to ",".
Before "." insert ", linkage association or construct association".

E4.1.6 reference
[434:2] Change "procedure name" to "procedure designator".
Reason: There are now procedure designators that are not simple names.

E4.2 Proposed new entries

E4.2.1 construct association

[427:26+] Insert definition of "construct association (16.4.1.5)":
"The association between the selector of an ASSOCIATE or SELECT TYPE construct and the associate name."

E4.2.2 linkage association
[431:33+] Insert definition of "linkage association (16.4.1.4)":
"The association between interoperable Fortran entities and their C counterparts."

E4.3 Other corrections

E4.3.1 associate name
[425:28-29] Replace the definition of "associate name" with "The name of the construct entity with which a selector of a SELECT TYPE or ASSOCIATE construct is associated within the construct."
Reason: This relates to a form of association, not "being known", which sounds like macro replacement. It is correct in the definition of "selector", and should be correct here.

E4.3.2 class
[426:19] Move this line to follow [426:43]
Reason: The term "class" is out of alphabetical order.

E4.3.3 dummy pointer
[428:39] Delete line.
Reason: The term "dummy pointer" is used only in section C.9.6, not in normative text, but appears in the glossary and the index. This gives undue prominence to an informal term.

E5. Index
E5.1 Index header
[555:5] Replace "primary or defining text" by "primary text or glossary definitions".

Rationale: Page numbers in bold face denote "primary or defining text"
but many such page numbers refer to the Glossary, which is informative and by definition cannot be "primary" or "defining". Moreover many terms in the index (action statement, allocatable variable, argument, associate name, etc, etc) refer only to the Glossary, not to normative text.

E5.2 BIND(C)
[66:8] and [77:20] Add index tag to "BIND"
Rationale: "BIND(C)" is not indexed where it is introduced.

E5.3 NAME=
[77:20] Add index tag to "NAME="
Rationale: "NAME=" is indexed only for its use in the INQUIRE statement, not for its use in the BIND specification.

E5.4 Ampersand
[395:27] Add index tag to "&".
Rationale: Ampersand is indexed as the free source continuation character and as a specifier in namelist input. Its use as a C operator in the definition of C_LOC in 15.1.2.5 is not indexed.

E5.5 Missing entries
[555-569] Three statements (SEQUENCE, USE and WAIT) are missing from the list of statements on page 567 but are present elsewhere in the index in the form "WAIT statement". Five further statements (ENUMERATOR, IMPORT, INTERFACE, PROGRAM, SUBROUTINE) are in the index only in the form "subroutine-stmt" and should ideally be both in the list on page 567 and present as entries in their own right. Full details of names and locations will be supplied separately if it is agreed in principle to upgrade the index in this way.

United States

General

Fix sloppy writing that was saying something different from what was intended

[75:26], [76:1] "same type" -> "same declared type" twice.
[75:26-27] Delete "For a polymorphic...declared type."

==========

Technical

Incorrect terminology

[14] In Table 2.1, "enumeration declarations" -> "enumeration definitions". In note 1 of Table 2.2, "enum statements" -> "enumeration definitions".

[45:17] Change "accessible" to "previously defined".

[291:21] Change "intrinsic function" to "intrinsic procedure"

[291:22] Change "intrinsic function" to "intrinsic procedure"

Fix problem with left tab limit and user-defined derived-type I/O (03-261r3)

[194:13-14] replace item (5) with
'(5) if the statement is not a child data transfer statement (9.5.3.7),

(a) position the file prior to data transfer (9.2.3.2), and
(b) for formatted data transfer, set the left tab limit (10.7.1.1).'

[194:28]
replace the final '.' with
', and for formatted data transfer, set the left tab limit (10.7.1.1).'

[235:32] insert 'nonchild' before 'data'

Example inconsistent with normative text (C1266)

[277:18+6] "REAL X" -> "REAL, INTENT(IN) :: X"

Inconsistent with intent, as evidenced by Note 16.8

[411:5] "; they" -> ", except that an accessed entity may have
the VOLATILE or ASYNCHRONOUS attribute even if the host entity
does not. The accessed entities"

Binding label fixes

[405:18-19] "procedure binding labels...BIND attribute" ->
"and entities with binding labels"
[405:22-23] "A binding label of a global entity" ->
"A binding label of an entity"
[405:23-24] "the binding label of any other global entity" ->
"the binding label of any other entity"
[405:24] "a name of any other global entity" ->
"the name of any other global entity"
{The unique name; not some possible local name.}
[405:25] "A global entity" -> "An entity"

==========

Editorial

Typographical errors

Change en-dash (LaTeX --) to minus (LaTeX $-$). (03-265)
[71:6+7], [213:21], [293:13,16], [502:9,10]

[269:Note 12.21] Dehyphenate "non-polymorphic" twice. (03-257)
[380:26] Delete spurious quote at the end of the line. (03-265)
[423:2] "variables" -> "variable". (03-257)
[453:2] "of MAR" -> "of MARY" (03-257)

Invalid C (03-257)

[486:35] Insert ";" after the last "c".

ISO Comments

		ed	Line numbers appear throughout the document.	Please remove.	
		ed	There are several instances of outdated text and other problems that could be solved by using the latest version of the ISO template.	Please use the ISO template, available online at http://isotc.iso.ch/livelink/livelink/fetch/2000/2123/SDS_WEB/sds_temp.htm.	
		ed	ISO International Standards should be identified as "International Standards"; standards of other organizations should be identified as "standards"; national standards should be identified as "national standards".	Please search the document and make the changes where appropriate.	
		ed	Incorrect text and format are used.	Please use the latest version of the Foreword, available from the ISO template.	
		ed	Incorrect headers are used.	Headers should read "ISO/IEC 1539-1:2004(E)".	
		ed	Incorrect footers are used.	Footers should read "© ISO/IEC 2004 — All rights reserved".	
		ed	***Only the first 14 pages reviewed (plus the introductory pages) as the electronic file was not available in the system.***		