To: J3
From: /interop
Subject: BINDNAME = and NAME= clarification
Resolution of unresolved items 150, second part of 151, 152, 153, 154, 156, 158

There are a number of unresolved issues with BINDNAME= and NAME=. Paper 99-210r1 attempted to summarize a discussion of these issues at meeting 150. Some of the edits to resolve these issues appeared in 99-205r2, edits for global data. This paper attempts to tie up the remaining issues.

Allow <language-binding-spec> (BIND) on a procedure declaration
[270:12+] add

"or <language-binding-spec>"

Allow only one BINDNAME= per procedure declaration (it can only resolve to one external name), disallow a <bind-spec> for procedure pointers (bind attribute belongs to the procedure associated with the pointer, not the pointer itself), and disallow binding names for dummy procedures (bind name comes from associated actual procedure).

Allow only one procedure to have a given NAME= or BINDNAME= <bind-spec>. Require an explicit interface for a procedure declared with a BIND attribute.
[270:23+] add

"Constraint: If <language-binding-spec> is specified, it shall contain at most one BINDNAME= <bind-spec>.
Constraint: If <language-binding-spec> is specified and <proc-entity> has either the POINTER attribute or is a dummy procedure, <language-binding-spec> shall not have a <bind-spec>.

Constraint: If a BINDNAME= or a NAME= <bind-spec> is present, <proc-decl-list> shall contain at most one <proc-decl>.

Constraint: If <language-bind-spec> is specified, the <proc-interface> shall be an <abstract-interface-name>, and <abstract-interface-name> shall be declared with a <language-bind-spec>."

NAME= is allowed to appear with BINDNAME= in procedure declarations, but as mentioned, a reference to a procedure may resolve to only one external entry name.

Disallow NAME= and BINDNAME= from appearing in an abstract interface body or an interface body for a dummy procedure.

[284:1-2] replace constraint with

"Constraint: A <bind-spec> shall not be specified in the <function-stmt> or <subroutine-stmt> of an abstract interface body (12.3.2.1) or an interface body for a dummy procedure."

[284:3-7] delete J3 note 150

[284:15-17] delete second part of J3 note 151. This is addressed by edits
to page 270 above.

Allow NAME= to be specified in a <prefix-spec> for a subprogram definition .
[284:18] delete constraint

Allow BINDNAME= to be specified in a <prefix-spec> of a procedure interface body. Allow only one BINDNAME= to be specified for a procedure interface body. (The processor can only resolve a procedure reference to one external bind name). Delete J3 note 152 [284:19-24] delete and replace with

"Constraint: If <language-binding-spec> is specified in the <prefix-spec> of a procedure interface body, <language-binding-spec> shall contain no more than one BINDNAME= <bind-spec>.

Note 12.xx
A subprogram definition with the BIND attribute is allowed more than one <bind-spec> in its <language-binding-spec>. This allows the subprogram to be referenced by more than one binding name should the processor have more than one companion processor (2.5.10), each with a different bind name mapping algorithm.

A <prefix-spec> for a procedure interface body shall not have more than one BINDNAME=<bind-spec>. The processor can resolve a reference of the procedure to at most one external bind name.
[End Note] "

Add note with example of procedure interface bodies with the BIND attribute.
[285:49+] add
"Note 12.xx

The following is an example of a procedure interface body with the BIND attribute.

```
USE ISO_C_BINDING

INTERFACE
   INTEGER(C_INT), BIND(C,NAME="FrEd") FUNCTION JOE (I, J, R)
        USE ISO_C_BINDING
        INTEGER(C_INT)  I, J
        REAL(C_FLOAT) R
  END FUNCTION JOE
END INTERFACE
```

The invocation of the function JOE

```
INT=JOE (1_C_INT, 3_C_INT, 4.0_C_FLOAT)
```

results in an reference to a function with a binding label "FrEd". "FrEd" may be a C function described by the C prototype

```
int FrEd (int l, int m, float x);
```

[End note]"

Unresolved issue 153 questions the usefulness of discussing binding labels for dummy procedures. Interface bodies and procedure declarations for dummy procedures prohibit a <bind-spec>, but do allow the BIND attribute. It is useful to indicate that the bind label for a dummy procedure with the BIND attribute comes from the associated actual procedure.

[289:5-14] delete section note 153

Unresolved issue 154 requests better definition of binding labels for C functions. Note 12.37 provides some examples of binding labels for Fortran procedures. An edit above adds a note to 12.2.5.1 with an additional example.

[289:16-22] delete J3 note 154

Unresolved issue 156 questions the usefulness of standardizing the spelling of BINDNAME= if its meaning is processor dependent. Note 12.38 describes a case where BINDNAME= is useful. While it's meaning is processor dependent, there are other features in Fortran with a standard spelling that are processor dependent (kind-type parameter values come to mind). Subgroup believes the benefits of a standard spelling of BINDNAME= outweigh the shortcomings of its meaning being processor dependent.

[290:2-12] delete unresolved issue 156

Subgroup note: need to discuss issue 158 page 290

Fix typo in Note 12.37
[289:35] change "use" to "to use"

Unresolved issue 158 questions the intent of the use of the word "may" in note 12.38. The use is not intended to grant permission, rather it is intended to reflect how a hypothetical processor might chose to implement names of global entities, or how another hypothetical processor might implement BINDNAME=. Edits are provided to change "may" to "might".

[290:14] change "may" to "might"

[290:17] change "may" to "might"

[290:23-33] delete unresolved issue 158