Subject: Issues 287, 288, 294, 296 and part of 290: Type-bound operators, assignment and generic procedures
From: Van Snyder
References: 00-304r1

# 1 Edits

Edits refer to 01-007. 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 (before) the indicated line. Remarks are noted in the margin, or appear between [ and ] in the text.

---

[Editor: after "(12.3.2.1)" insert ", or a subroutine and a generic procedure binding (4.5.1.5),"] 31:18

---

[Editor: "*construct*" ⇒ "*stmt*" twice.] 41:15-16

---

R440 *proc-binding-stmt* **is** *specific-binding* 41:20-21
**or** *generic-binding*
**or** *final-binding*

[Editor: Move [41:33-35] to here.]

---

R441 *specific-binding* **is** PROCEDURE [ ( *abstract-interface-name* ) ] ■ 41:22

---

[Editor: Move [41:36] to here.] 41:23+

Constraint: The *abstract-interface-name* shall be specified if and only if the *binding* is NULL() and is not overriding an inherited binding. [This was formerly at [41:37-38].]
[Now that we have generic bindings, if may be desirable to change this constraint, to avoid a disturbing antisymmetry: One (presently) *cannot* specify the interface of a deferred specific binding that is overriding another specific binding, but one is *required* to specify the interface of a deferred generic binding that is overriding a generic binding – otherwise we wouldn't know which one to override.]

[Editor: Move [42:28-29] to here.]

---

R441$\frac{1}{3}$ *generic-binding* **is** GENERIC [ ( *abstract-interface-name* ) ] ■ 41:24-26
■ [, *binding-attr-list*] :: *generic-spec* => *binding-list*

Constraint: The *abstract-interface-name* shall be specified if and only if *binding-list* is a single binding which is NULL().

Constraint: If *generic-spec* is *generic-name*, *generic-name* shall not be the name of a specific binding of the type.

Constraint: If *generic-spec* is OPERATOR ( *defined-operator* ), the interface of each binding shall be as specified in 12.3.2.1.1.

Constraint: If *generic-spec* is ASSIGNMENT ( = ), the interface of each binding shall be as specified in 12.3.2.1.2.

Constraint: If *generic-spec* is *dtio-generic-spec*, the interface of each binding shall be as specified in 9.5.4.4.3. The type of the **dtv** argument shall be *type-name*.

R441$\frac{2}{3}$ *final-binding* **is** FINAL [ :: ] *final-subroutine-name-list*

---

[Editor: Delete unresolved issue note 287.] 41:27-32

---

[Editor: Delete. Note that [41:33-36] have been moved upward.] 41:37-40

---

[Editor: Delete unresolved issue note 288.] 42:1-11

---

[Editor: Delete.] 42:12-17

---

Constraint: In a *generic-binding*, if *generic-spec* is *dtio-generic-spec*, PASS_OBJ shall not be specified. 42:34-35

Constraint: In a *generic-binding*, if *generic-spec* is OPERATOR ( *defined-operator* ) or ASSIGNMENT ( = ), PASS_OBJ shall not be specified, but each interface shall satisfy the conditions that would be required if it had been specified.
[**Alternative** that Malcolm prefers:] In a *generic-binding*, if *generic-spec* is OPERATOR ( *defined-operator* ) or ASSIGNMENT ( = ), PASS_OBJ shall be specified.

Constraint: If PASS_OBJ is specified, the procedure named by *procedure-name* or the interface named by *abstract-interface-name* shall have a scalar nonpointer nonallocatable dummy argument of type *type-name*. The first such dummy argument shall be polymorphic if and only if the type is extensible.

Constraint: PASS_OBJ shall be specified for an overriding binding if and only if it is specified for the binding being overridden.

Constraint: PASS_OBJ shall be specified for a *generic-binding* if and only if it is specified for all generic bindings, both inherited and declared within the type definition, with the same *generic-spec*.

Constraint: NON_OVERRIDABLE shall be specified for a *generic-binding* if and only if it is specified for all generic bindings with the same *generic-spec* declared within the type definition.

Constraint: Within the *specification-part* of a module, each *generic-binding* shall specify the same accessibility, either explicitly or implicitly, as every other *generic-binding* in the same type definition that has the same *generic-spec*.

---

**J3 internal note** 42:37+

Unresolved issue xxx
Some time ago we changed the syntax for deferred bindings from PROCEDURE :: *binding-name* => NULL( *abstract-interface-name* ) to PROCEDURE( *abstract-interface-name* ) :: *binding-name* => NULL(), partly to make it consistent with PROCEDURE statements, and partly to avoid introducing the use of *abstract-interface-name* as an actual argument. Now that we have generic bindings, should we (almost) change this back? The "almost" part means to allow NULL( *procedure-pointer* ) instead of NULL( *abstract-interface-name* ), or the present form, but not both in the same statement.

---

[Editor: Delete "If ... extensible." It's covered by one of the constraints inserted by the edit for [42:34-35] above.] 42:39-42

---

[Editor: Delete.] 42:45-43:2

---

[Editor: "-*construct*" ⇒ "-*stmt*".] 47:12

---

There are five categories of type-bound procedures: A binding specified by PROCEDURE specifies a named type-bound procedure. A binding specified by GENERIC :: *generic-name* specifies a named type-bound generic interface. A binding specified by GENERIC :: OPERATOR ( *defined-operator* ) specifies a type-bound defined operation (12.3.2.1.1). A binding specified by GENERIC :: ASSIGNMENT ( = ) specifies a type-bound defined assignment (12.3.2.1.2). A binding specified by GENERIC :: *dtio-generic-spec* specifies a type-bound user-defined derived-type input/output procedure (9.5.4.4.3, 12.3.2.1.3). 47:12+ Same ¶

[Editor: Add the above paragraph to the index for "generic interface".]

The interface of a binding is the interface of the procedure specified by *procedure-name* or the interface of the abstract interface specified by *abstract-interface-name*.

A binding is associated with a derived-type definition if it is specified within that definition, or inherited into the type that definition defines.

---

A binding specified by GENERIC may override one inherited from the parent type (4.5.3.2) with the same *generic-spec*. Otherwise it extends the generic interface for that *generic-spec*. If the *generic-spec* is *generic-name*, the set of all accessible nonoverridden bindings inherited from the parent for that *generic-spec*, together with the bindings for that *generic-spec* declared within the type definition, constitutes a type-bound generic interface. If the *generic-spec* is not *generic-name*, the set of all accessible nonoverriding generic bindings for that *generic-spec*, together with specific procedures specified in accessible generic interface blocks with the same *generic-spec*, constitute a generic interface. Generic interfaces specified in these ways shall satisfy the requirements stated in 14.1.2.3. 47:25+ New ¶

The same *generic-spec* may be used in several *generic-binding*s within a single derived-type definition. All bindings specified with a particular *generic-name* within a single derived-type definition contribute to the same type-bound generic interface. Otherwise, all bindings specified with a particular *generic-spec* contribute to the same generic interface.

---

[Editor: Delete.] 47:31-38

---

For purposes of overriding (4.5.3.2) and generic resolution (14.1.2.3, 14.1.2.4.1$\frac{1}{2}$), the declared type of a passed-object dummy argument (4.5.1.7) of a procedure binding inherited from the parent type is considered to be the same as the type into which it is inherited. 52:20+
[This avoids saying "Except for the passed-object dummy argument," and then needing to say what rules apply to the passed-object dummy argument, at [53:8] and in (14.1.2.3, 14.1.2.4.1$\frac{1}{2}$).]

---

A procedure binding declared within a derived type definition **overrides** one inherited from the parent type if: 52:41-53:3

(1) It is declared using PROCEDURE and it has a binding name that is the same as one inherited from the parent type, or

(2) It is declared using GENERIC, it has the same *generic-spec* as one inherited from the parent type, and the interface of the procedure specified by the *procedure-name* or the interface specified by *abstract-interface-name* is not distinguishable, by using the rules in 14.1.2.3, from one inherited from the parent for the same *generic-spec*.

If it is declared using GENERIC ( *generic-name* ) but it does not override one inherited from the parent, it extends the type-bound generic interface having that *generic-name*. Otherwise if it is declared using GENERIC but it does not override one inherited from the parent, it extends the generic interface having that *generic-spec*. A binding that overrides one inherited from the parent shall match the overridden binding in the following ways:

---

[Editor: Delete "the procedure of" and "that of".] 53:4

---

[Editor: "Except ... argument, the" ⇒ "The". The deleted part is now provided by the edit at [52:20+] above. This is defective as it presently stands because it doesn't say anything about how the characteristics of the passed-object dummy arguments of the inherited and overriding binding correspond.] 53:8

---

| | |
|---|---|
| [Editor: Delete.] | 53:34-38 |
| [Editor: After "(12.3.2.1)" insert ", or a generic procedure binding (4.5.1.5),"] | 56:29 |
| [Editor: "block (" ⇒ "(4.5.1.5,".] | 112:16 |
| [Editor: Delete "block" thrice.] | 112:18-22 |
| [Editor: Delete unresolved issue note 294.] | 114:1-5 |
| [Editor: "block (" ⇒ "(4.5.1.5,".] | 126:35 |
| [Editor: "block (" ⇒ "(4.5.1.5,".] | 127:5 |
| [Editor: Delete "block".] | 130:2 |
| [Editor: "block (" ⇒ "(4.5.1.5,".] | 130:16 |
| [Editor: "block (" ⇒ "(4.5.1.5,".] | 132:26 |
| [Editor: "interface block" ⇒ "generic interface".] | 247:27-28 |
| [Editor: "it ... names" ⇒ "the functions shall have interfaces that are distinguishable according to the rules stated in 14.1.2.3, and references are resolved using the rules stated in 14.1.2.4.1". | 248:15 |
| If ASSIGNMENT ( = ) is specified in a generic specification, all of the procedures in the generic interface [NOTICE that the word "block" is intentionally deleted!] | 248:28 |
| [Editor: Before "Each" insert "A particular defined assignment may, as with generic names, apply to more than one subroutine, in which case the subroutines shall have interfaces that are distinguishable according to the rules stated in 14.1.2.3, and references are resolved using the rules stated in 14.1.2.4.1."] | 248:29 |
| [Editor: "As ... name" ⇒ "A particular *dtio-generic-spec* may, as with generic names, apply to more than one subroutine, in which case the subroutines shall have interfaces that are distinguishable according to the rules stated in 14.1.2.3, and references are resolved using the rules stated in 14.1.2.4.3."] | 249:13-15 |
| [Editor: Delete unresolved issue note 296.] | 249:16-25 |
| [Editor: Delete – redundant now that "distinguishable according to the rules stated in 14.1.2.3" is specified above.] | 249:26-31 |
| R1214$\frac{1}{2}$ *abstract-interface-name* **is** *name*<br>Constraint: The *name* shall be the name of an abstract interface (12.3.2.1). | 250:43 |
| [Editor: After "names" add "or generic bindings (4.5.1.5)".] | 342:14 |
| [Editor: "procedure name" ⇒ "identifier".] | 344:21 |
| [Editor: After "reference" insert ", is a generic type-bound reference".] | 344:23 |
| [Editor: After "reference" insert ", is a specific type-bound reference, is a reference to a user-defined derived-type input/output procedure".] | 344:24 |
| [Editor: "A procedure name" ⇒ "An identifier".] | 344:25 |
| [Editor: "an interface block with that name" ⇒ "a generic interface with a *generic-spec* other than a *dtio-generic-spec* that specifies that identifier".] | 344:26 |
| [Editor: "name" ⇒ "identifier" twice.] | 344:27-28 |

[Editor: "procedure name" ⇒ "identifier".] 344:29

[Editor: "name" ⇒ "identifier".] 344:30

[Editor: "name" ⇒ "identifier" twice.] 344:32-33

(1$\frac{1}{2}$) A reference is established to be to a generic type-bound interface if it is of the form *data-ref % binding-name* and *binding-name* is the *generic-name* in a *generic-spec* in a binding specified using GENERIC within the definition of the declared type of *data-ref*. 344:34+

[Editor: "names" ⇒ "identifiers".] 345:3

(1) If the declared types of the actual arguments that are not procedures or procedure pointers are type compatible with, and have the same kind type parameters as the corresponding dummy arguments of one of the nonoverridden specific interfaces of a generic interface 345:4

[Editor: "name" ⇒ "identifier".] 345:5

[Editor: "interface block that provides that" ⇒ "generic interface that provides that specific".] 345:7

[Editor: "procedure" ⇒ "interface".] 345:8

[Editor: "name" ⇒ "identifier".] 345:10

[Editor: "interface block that provides that" ⇒ "generic interface that provides that specific".] 345:12-13

[Editor: "name" ⇒ "identifier".] 345:42

[Editor: "name is a function name or subroutine name, the name" ⇒ "identifier refers to a function or subroutine, the identifier".] 345:44-45

If the procedure has a passed-object dummy argument, the reference is to the procedure named by the *procedure-name* specified by a *proc-binding-stmt* that is associated with (4.5.1.5) the dynamic type of the actual argument that corresponds to the passed-object dummy argument. NULL() shall not be specified in that *proc-binding-stmt*. 345:46+ New ¶

[Editor: Add a new section, after – not a subsection of – 14.1.2.4.1.]

**14.1.2.4.1$\frac{1}{2}$ Resolving references to type-bound generic procedures**

If a type-bound generic interface is specified by *data-ref % binding-name* in a function reference or call statement:

(1) If the reference is consistent with one of the specific interfaces in the generic binding associated with (4.5.1.5) the declared type of the *data-ref* and having a *generic-name* that is the same as the *binding-name*, that interface is selected.

(2) Otherwise, if the reference is consistent with an elemental reference to one of the specific interfaces in the generic binding associated with the declared type of the *data-ref* and having a *generic-name* that is the same as the *binding-name*, that interface is selected.

(3) If an interface is selected by (1) or (2) above, the reference is to the specific binding associated with the dynamic type of the *data-ref* that either is the binding that provides the specific interface in (1) or (2), or overrides it.

If no interface is selected in (1) or (2), or the specific binding determined in (3) is a deferred binding (4.5.1.5), an error condition occurs.

($2\frac{1}{3}$) A reference is established to be to a specific type-bound procedure if it is of the form *data-ref % binding-name* and *binding-name* is the *binding-name* in a *specific-binding* within the definition of the declared type of *data-ref*. 345:48+

($2\frac{2}{3}$) A reference is established to be to a user-defined derived-type input/output procedure by the rules specified in 9.5.4.4.3.

---

($5\frac{1}{2}$) If the reference is of the form *data-ref % binding-name* and *binding-name* is the same as one specified in a *specific-binding* associated with (4.5.1.5) the declared type of *data-ref*, then the reference is to the *procedure-name*, if any, specified in the *specific-binding* associated with the dynamic type of *data-ref* and that has the same *binding-name*. If that *specific-binding* is a deferred binding (4.5.1.5), an error condition occurs. 346:20+

---

[Editor: Delete unresolved issue note 290.] 346:24-27

---

**deferred procedure binding** (4.5.1.5): a type-bound procedure binding that specifies the NULL() intrinsic. A deferred procedure binding shall not be invoked. 402:14+

---

[Editor: "an interface block" ⇒ "a generic interface".] 402:23

---

[Editor: "an interface block" ⇒ "a generic interface".] 403:10

---

**generic identifier** (4.5.1.5, 12.3.2.1): A lexical token that appears in a generic procedure binding or an INTERFACE statement and is associated with all the procedure names or abstract interfaces in the generic interface. 404:15-17

**generic interface** (4.5.1.5, 12.3.2.1): An interface specified by a generic procedure binding or a generic interface block.

---

**generic procedure binding** (4.5.1.5): a procedure binding specified by a GENERIC procedure binding statement. 404:18+

---

[Editor: After "type" insert ", as a defined operator, or by defined assignment".] 409:31

---

[Editor: After "module" insert ", or a generic procedure binding may be specified in a derived type definition".] 445:4