Subject: Derived-type input/output and its relation to type-bound procedures
From: Van Snyder
References: 00-186

1 Introduction

There are several problems with derived-type input/output that could be addressed by a minor extension to the type-bound procedure mechanism: It is possible to access a type from a module, but not access its input/output procedures, and it is not possible to inherit, override or defer a derived-type input/output procedure.

This paper depends on paper 00-186. It should be processed, if at all, after that paper passes, or another one that specifies how kind type parameters interact with type-bound procedure invocation passes.

2 Proposed change

Replace the interface-block-based mechanism to specify derived-type input/output procedures by a variation on the type-bound procedure declaration mechanism, e.g.

```
PROCEDURE, READ(FORMATTED) => my_read_routine_formatted
```

serves the same purpose as the interface block at [190:41]. Obviously similar extensions provide for unformatted input and for output. This binding cannot be excluded during use association, is inherited into extension types, can be overridden in them, and an obvious variation allows deferred derived-type input/output procedures to be specified.

The advantage of this approach is that it solves the problems noted above. The disadvantage (or another advantage, depending on your point of view) is that it does not allow a derived-type input/output procedure to be a dummy argument or a procedure pointer.

3 Edits

Edits refer to 00-007r1. 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 the indicated line. Remarks for the editor are noted in the margin, or appear between [and] in the text.

---

or PROCEDURE (*proc-interface-name*), ■ 44:18+
■ *dtio-binding-attr* => NULL()
or PROCEDURE, *dtio-binding-attr* => ■
■ *dtio-binding-list*

[Editor: Add to the constraint:] 44:25
If *proc-interface-name* and *dtio-binding-attr* are both specified, the interface shall specify a subroutine having an explicit interface as specified for the same *dtio-binding-attr* in section 9.5.4.4.3.

R442b *dtio-binding-attr* **is** READ(FORMATTED) 45:3+

or READ(UNFORMATTED)
or WRITE(FORMATTED)
or WRITE(UNFORMATTED)

R442c *dtio-binding* **is** *procedure-name*
or NULL(*procedure-name*)
or NULL(*procedure-pointer-name*)

Constraint: The *procedure-name* shall specify an accessible module procedure or external procedure. The *procedure-pointer-name* shall specify an accessible procedure pointer. The *procedure-name* or *procedure-pointer-name* shall have an explicit interface as specified for the same *dtio-binding-attr* in 9.5.4.4.3.

Constraint: If several specific or deferred (4.5.1.5) procedures are specified for a single *dtio-binding-attr*, their interfaces shall differ as specified in 14.1.2.3.

Note 4.19$\frac{1}{2}$

The interfaces are specified nearly completely in section 9.5.4.4.3. The only latitude for differences is that, for a particular type and *dtio-binding-attr*, the derived type dummy arguments can have different kind type parameters.

4.5.1.5.2 Derived-type input/output subroutine 49:30+

A procedure binding with a *dtio-binding-attr* specifies a user-defined derived-type input/output subroutine. Its use and the characteristics it shall have are described in 9.5.4.4.3. The set of user-defined derived-type input/output subroutines that are bound to the type or that are inherited (4.5.3.1) from the parent type and not overriden (4.5.3.2) is a generic interface.

[Editor: Insert a new paragraph, not within note 4.44:] 54:45+
If a procedure binding declared in a type definition has the same *dtio-binding-attr* and the same kind type parameters for the derived-type argument as one inherited from the parent type then the binding declared in the type overrides the one inherited from the parent type. Otherwise it extends the generic interface for the declared type and specified *dtio-binding-attr*.

[Editor: Replace "any procedure ... matches" by "an external or module subroutine, bound to the type by a procedure binding with a *dtio-binding-attr* as specified in section 4.5.1. Its interface shall match".] 189:26

[Editor: Replace "procedure" by "subroutine".] 189:28

[Editor: After "transferred" insert ", as described in 14.1.2.4.2$\frac{3}{4}$".] 189:30

[Editor: Replace "When an interface ... scoping unit" by "If a derived-type input/output procedure is selected as specified in the previous paragraph".] 189:31-32

(1) If the *dtio-binding-attr* is READ (FORMATTED): 190:26

(2) If the *dtio-binding-attr* is READ (UNFORMATTED): 190:41-42

(3) If the *dtio-binding-attr* is WRITE (FORMATTED): 191:2-3

(4) If the *dtio-binding-attr* is WRITE (UNFORMATTED): 191:18-19

[Editor: Delete.] 191:30

[Editor: Delete.] 246:19-22

[Editor: Delete.] 249:18-20

[Editor: Delete.] 251:11-19

14.1.2.4.2$\frac{3}{4}$ Resolving derived-type input/output procedure references 346:22+

The *effective set of input/output procedures* for a type and *dtio-binding-attr* is the set of procedures inherited for that *dtio-binding-attr* from the parent of the type, minus the overridden ones, plus the ones declared in the type. Each procedure in an effective set has a corresponding one in each effective set for each extension type – either the same procedure or one that overrides it. Each effective set is a generic interface.

A derived-type input/output procedure for one of the four kinds of data transfer specified in 9.5.4.4.3 and a particular type of list item is selected as follows:

(1) At most one procedure is selected from the effective set of procedures for the *dtio-binding-attr* and the declared type of the list item, according to the generic resolution rules (14.1.2.4.1).

(2) If a procedure is selected in step (1), the reference is to the procedure from the effective set, for the *dtio-binding-attr* and the dynamic type of the list item, that corresponds to the procedure selected in step (1). Otherwise, intrinsic input/output is used.

If the reference is to a deferred binding, an error condition occurs.