To: J3
Subject: The EXTERNAL attribute.
From: Malcolm Cohen.
Date: 2002-08-13

1. Introduction

The EXTERNAL attribute definition is both badly written and broken. This is a shame, because the F95 version is less badly written (though not perfect) and not broken. Somewhere, somehow, a wheel has fallen off.

2. What is wrong

The major thing that is wrong is that 5.1.2.6 prevents access to intrinsic procedures. All procedures referenced, if they aren't explicitly specified to have the EXTERNAL attribute, are given it implicitly! (Even if the user says INTRINSIC, it still gets the EXTERNAL attribute!!!)

Secondary things are:
- the text claims that names have attributes
- the text duplicates definitions elsewhere

Finally, there has been a language change. The previous requirement for procedures as actual arguments needing to be explicitly given the EXTERNAL attribute has disappeared. I have not managed to find any evidence of this having been deliberately done, and plenty of evidence to support the theory that it was inadvertant.

3. Language change discussion

In 98-144 (ok, so I'm a bit far behind in my reading), the editor says:

> While on the subject, since this looks like a complete list of ways
> that a name can get the external attribute, I added a sentence about
> getting the attribute implicitly. It seems to me that otherwise
> the list is misleading in its incompleteness. But...
>
> ***ACTION FLAG***
> We badly need an xref for how a name can get the external attribute
> implicitly. I wasn't sure where the bext xref was and didn't take
> the time to find one. Left a J3note there instead.
> ***END ACTION

Bad Richard. There was no implicit getting of the EXTERNAL attribute in previous Fortran standards. You seem to have misinterpreted the "EXTERNAL attribute specifies that the name is that of an external procedure..." as meaning that if the name is that of an external procedure, it has the EXTERNAL attribute. Well, that is certainly untrue in F95. E.g.

> "If such a conflict occurs and involves the name of an external procedure,
> the processor is permitted to use the intrinsic procedure unless the name is
> given an interface body or the EXTERNAL attribute in the same scoping unit
> (14)."

Note the lack of "explicitly given" in front of "the EXTERNAL attribute".

Then, in paper 98-200, the editor says:

> The following miscellaneous edits are proposed. The editor does not
> believe that any of these edits make technical changes. Some of

them are from paper 98-180, after discussion.
...
[65:14-17] Replace "The EXTERNAL attribute may also be implicitly specified (???)." and the J3 note by a separate para reading

"Any name that is used in a scoping unit as the <function-name> of a <function-reference> or as the <subroutine-name> of a <call-stmt> implicitly acquires the external attribute in that scoping unit if it is not the name of an accessible statement function or internal procedure, is not accessed by host or USE association, and is not explicitly given the external attribute."

The editor was mistaken. 98-121r1 altered the actual argument requirements to be
"If a name has the EXTERNAL attribute ... it may be used as an actual argument ..."
i.e. it omitted the "explicitly given" requirement. This had had no effect until the implicit EXTERNAL stuff was added.

If there was a technical proposal which blessed this apparently inadvertant alteration of our semantics, I've not spotted it (and I looked fairly carefully). Therefore I'm going to propose that we revert to the F95 situation where the EXTERNAL attribute must be explicitly specified in order to pass something as an actual argument. Since being the target of a procedure pointer assignment is like being passed as an actual argument, I'll propose extending this requirement to that case as well.

It doesn't necessarily matter if we have some sort of "implicit EXTERNAL", as long as it is described correctly (unlike now). But seeing no advantage in specifying it like this, I'll propose simply deleting it.

4. Implicit procedure pointers

We now appear to have the situation where

```
POINTER P
...
PRINT *,P()
```

makes P into a procedure pointer, but

```
POINTER P
REAL,EXTERNAL :: MYFUN
...
P => MYFUN
```

does not.

This is silly. In fact, the whole idea is bad: if people want procedure pointers, then THEY SHOULD NEED TO DECLARE THEM EXPLICITLY. To do otherwise is to create compiler confusion complexity and inconsistencies, and to inhibit the detection of simple user errors. Therefore I am proposing a TECHNICAL CHANGE: that procedure pointers must be explicitly given the EXTERNAL attr.

5. Procedure pointers missing from classification

In 12.1.2 "Procedure classification by means of definition", we omit any explicit mention of procedure pointers. Dummy procedure pointers fall under the "dummy procedure" classification, but non-dummy ones escape completely.

This is probably a problem when we blithely say "procedure pointers are procedures with the POINTER attribute": currently they arguably aren't properly procedures because they are missing from the classification.

6. Miscellaneous

It would be nice to come right out and say that we have dummy procedure pointers in the right place, rather than just using the term elsewhere.

The language used to define dummy procedures is a bit vague ("specified as a procedure"). It would be easier to understand if it said "specified to be a procedure" instead.

7. Edits to 02-007r2

[81:13] Change "a name" to "an entity".

[81:14] Delete ", abstract interface"
{They cannot appear here any more (250r1).}

[81:14-17] Change "be explicitly ... body" to
"also be specified by an EXTERNAL statement (12.3.2.2), a <procedure-declaration-stmt> (12.3.2.3) or an interface body that is not in an an interface block (12.3.2.1)."
{Just give the list of other places that give the attribute, with references. We don't need to give the exact syntax - that is done in the other places.}

[81:18-21] Delete.
{Unnecessary and wrong.}

[81:22-23] Delete.
{This is nonsense. Saying "REAL,EXTERNAL :: F" doesn't make F into a function, because this is not the definition of F. It's only standard conforming if F actually *IS* a function, but that is a horse of a different colour. The only thing that makes F into a function is the definition of F.}

[81:24-26] Replace with
"If an external procedure or dummy procedure is used as an actual argument or is the target of a procedure pointer assignment, it shall be declared to have the EXTERNAL attribute."
{Reinstate requirement for actual arguments.
TECHNICAL CHANGE: Impose requirement for being a <proc-target>.}

[81:27-39] Replace with
"A procedure that has both the EXTERNAL and POINTER attributes is a procedure pointer."
{TECHNICAL CHANGE: See above.}
{We don't have to say here that a dummy procedure is a procedure that is a dummy argument - that is already said elsewhere (indeed, in the place where we *DEFINE* what the term "dummy procedure" means).}
{Cut the verbiage. Also cut out incorrect assertions (viz [81:30]). Paper 250r1 has already dealt with abstract interfaces.}

[253:25] After "dummy procedure" insert
"(which may be a dummy procedure pointer), a nondummy procedure pointer,"
{Include procedure pointers in our classification. We have to say "nondummy" because the sentence is an exclusive-or list.}

[254:7] Change "as" to "to be".
{Make dummy procedure definition easier to understand.}

[254:8] Append
"A dummy procedure with the POINTER attribute is a dummy procedure pointer."
{Admit to the existence of dummy proc ptrs as a subclass of dummy procedures.}

[254:8+] Insert new section

"12.1.2.3A Procedure pointers

A procedure pointer is a procedure that has the EXTERNAL and POINTER attributes; it may be pointer associated with an external procedure, a module procedure, an intrinsic procedure, or a dummy procedure that is not a procedure pointer. A procedure pointer may be a dummy argument, a structure component, or a local entity."
{Classify procedure pointers. We don't need much here.}

[257:20] Delete the "\" characters before the "{" and "}" characters that are delimiters for the "\mindex" macro's argument.
{As it stands, it puts "{" and "}" into the index.}

[262:39-40] Change "The PROCEDURE statement" to "It", and move the sentence to [261:14] - append to the paragraph.
{This should be specified up front, not as an afterthought.}

[392:15-16] Delete "If ... argument."
{The BIND attribute was deleted from the list of characteristics of procedures by 166r2. It also does not make sense for a dummy argument to have a binding label.}

[394:16] Replace "in the case of generic names. A" by "that a".