Derived Type Encapsulation X3J3/97-145

Problem:

It is possible to define an opaque derived type OT that has a user-defined assignment procedure. However, the user-defined assignment procedure is not invoked when assigning objects of another type T containing components of type OT. This applies both to pointer and non-pointer components. Since the very reason type OT has user-defined assignment is probably that the intrinsic assignment was inappropriate, this means that an inappropriate assignment will be performed unless the writer of type T takes special care (i.e. also has user-defined assignment).

Example:

```
MODULE iso_varying_string
        TYPE varying_string
                PRIVATE
                CHARACTER, POINTER :: value(:)
        END TYPE
        INTERFACE ASSIGNMENT(=)
                MODULE PROCEDURE assign_vs_to_vs
        END INTERFACE
CONTAINS
        SUBROUTINE assign_vs_to_vs(var,expr)
                TYPE(varying_string),INTENT(OUT) :: var
                TYPE(varying_string),INTENT(IN) :: expr
                ALLOCATE(var%value(SIZE(expr%value,1)))
                var%value = expr%value
        END SUBROUTINE
END MODULE
PROGRAM programme
        USE iso_varying_string
        TYPE mytype
                TYPE(varying_string) name
        END TYPE
        TYPE(varying_string) x, y
        TYPE(mytype) a, b
        x = y ! invokes "assign_vs_to_vs(x,y)"
        a%name = b%name ! invokes "assign_vs_to_vs(a%name,b%name)"
        a = b ! does intrinsic assignment on A%name and B%name
END
```

Other Problems:

(1) It is very easy to drop the user-defined assignment even for normal variables - a simple "USE iso_varying_string,ONLY:varying_string" quietly loses the appropriate assignment.

(2) Deliberate subversion of the "opaque" data type is just as easy.

Solution:

1. Add "default assignment" that is used by the assignment statement for derived types. Default assignment will

(i) do pointer-assignment on pointer components

(ii) do intrinsic assignment on non-pointer components of intrinsic type

(iii) do defined assignment on non-pointer components of derived type if there is a defined assignment procedure for the component type; and

(iv) do default assignment on other non-pointer components of derived type.

And, add a note to section 1.x noting that this is a change from F95 to F2002.