Subject: Specifications, syntax and semantics for type-bound operators
From: Van Snyder

1 Specifications

1. Bind operators and assignment to types, creating or extending a generic interface.
2. Perform generic resolution using the declared types of the operands.
3. Dispatch to an overriding binding using the dynamic type.

2 Syntax

Bind operators and assignment to a type by using

proc-binding-stmt **is** GENERIC, *binding-attr-list* :: ■
■ *binding-generic-spec* => *binding-list*
or GENERIC (*proc-interface-name*) ■
■ , *binding-attr-list* :: ■
■ *binding-generic-spec* => NULL()

binding-attr-list **is** < as at present >

Constraint: PASS_OBJ shall be specified.

binding-generic-spec **is** OPERATOR (*defined-operator*)
or ASSIGNMENT (=)

3 Semantics

All of the procedures and NULL() bindings for assignment or a particular operator constitute a type-bound generic interface. All of the accessible type-bound generic interfaces, and accessible free (non-type-bound) generic interfaces, for assignment or for a particular operator, constitute a single generic interface. The bindings, and specific procedures, of this generic interface, taken as a whole, shall be distinguishable by the rules in subclause 14.1.2.3.

The requirements for arguments shall be the same as for generic operators or assignment.

A binding overrides one inherited from the parent type if the requirements in subclause 4.5.3.2 are satisfied. Otherwise it extends the generic interface.

Where a defined operator or assignment appears, all of the accessible generic interfaces for that operator or assignment, both type-bound and free, are considered. A specific procedure is selected by the usual rules for generic resolution in subclause 14.1.2.4. If there is an overriding binding for the dynamic type of the passed-object argument, the overriding binding is selected.

In any case, if a NULL() binding is selected, an error condition occurs.