6. CLASS DECLARATION

A class declaration defines the class associated with a class identifier. Instances of a the class may be created dynamically. The formal parameters, virtual quantities and local quantities are called the "attributes" of the class. The statements within the class are called the "operation rule" of the class.

The end of the operation rule of a class is indicated by a call on the end class body (ECB) subroutine which will return following inner in the prefix (if any).

```
ref (object) procedure ECB(p); ref (prototype) p;
   begin ref (program) out; ref (driver) x;
      procedure delete;
         begin x.rp := false;
           if x.obj. PP.local classes then
           begin x.drex :- x.drp; x.pex :- none; x.acs :- none; end
           else begin x.obj.MDP :- none; deletenotice (x) end
         end delete;
      if p.plev ≠ 0 then go to p.prefix [p.plev-1].inretur;
      x :- CD;
      if CD.rp and not CD.pb then
      begin CD :- CD.drp; delete; go to L2;
      L1:   CD :- CD.drp;
      L2:   while not CD.rp do CD :- CD.drex;
            if not CD.pb then go to L1;
            x :- CD.drp;
            while not X.rp do x :- x.drex;
            x.drex :- CD;
      L3:   if CD.pex =/= none then go to L4;
            CD :- CD.drex;
            go to L3;
      L4:   out :- CD.pex;
      end else
```

```
      begin
          if x.pb then
          begin CD :- x.drp;
             out :- x.obj.PP.endblk;
             deletenotice (x);
             go to ud
          end else
          begin out :- x.pex;
             CD :- CD.drex;
             ECB :- x.obj;
             restore (x.acs);
             delete
          end
       end;
       update display;
       ud:   go to out;
end ECB;
```

If this class has a prefix, return after the statement inner; in the prefix.

The compiler may determine this. In that case, return is compiled directly into the class body and ECB is not entered.

If the object has local class declarations, the master driver is made special by setting "pex" to none and the dynamic link (drex) equal to the static link (drp).

Note that "md" must not be changed in this case since the driver must remain a master driver.

If the object has no local class declarations, the driver is deleted (put on the available storage chain for POOL 2) by a call on "deletenotice".

The accumulator stack is restored. The function value of this procedure is a pointer to the B.I., except for a prefixed block or a detached object when it is none.

DISPLAY is updated.

6.1 Subclasses

In the declaration of a class D, the class may be prefixed by another class C. This defines D as a subclass of C. C may itself have a prefix. The sequence of prefixes A, ..., C is called the prefix chain of D. The prefix chain for a class C may not include a subclass of C.

The prefix to a class C is limited to be either a system defined class or a class declared within the same block as the declaration of C.

Declarations and operation rules of a class and its prefix are concatenated according to the rules given in the Common Base.

At runtime, the fact that the class D has the class C as prefix may be indicated by a pointer, "prefix", from the prototype of D to the prototype of C.

Since the declarations D1,.....,Dn and statements S1, ...,Sn in the prefix sequence P1,.....,Pn-1 of the class Pn by definition should be executed in the order first declarations, then statements, it is advantageous to have a pointer from the prototype of Pn to the prototype of each of the classes in the prefix hierarchy.

A class Pj may have a "split body", which explicitly indicates that if the object is of a class Pn which is a subclass of Pj, the concatenated operation rule of Pj+1, ...,Pn should be executed prior to continuing the operation within Pj. A split body of a class Pj is

indicated by the occurrence of the statement inner; in the declaration of Pj. A class declaration with no inner; statement should be regarded as having an implicit inner; at the end of the operation rule.

When Pj+1 has Pj as prefix, the return point at the end of the operation rule of Pj+1 should be after the inner; statement of Pj. It is worth giving the compiler some intelligence at this point. At the end of the operation rule of the Pj+1, an unconditional jump could be compiled to the statement following inner; in the innermost of the classes P1,.....,Pj which has a split body. If neither of these have a split body, a call on (a slightly simplified) ECB subroutine is compiled which in fact indicates the end of the concatenated operation rule.

The formal definition here is written assuming no such compiler intelligence.

A call on the call inner (CINNER) subroutine represents the explicit or implicit inner statement.

```
procedure CINNER(lev); integer lev;
      begin ref (prototype) p;
         p :- CD.obj.PP.prefix[lev+1];
         if p =/= none then
         go to p.statements
end CINNER;
```

6.2 Parameters

The parameters given when an instance of a class is generated are matched against the formal parameters as described for procedures. The formal parameters must be specified, and permissible specifications are type and type arrays. The type of an actual parameter must be compatible to the type specified for the formal parameter.

Parameters of types integer, real, Boolean and character are by definition called by value, while ref and text parameters are called by "copy". Arrays are called by "copy description". The option exists for the user to specify text and arrays as being called by value.

It is suggested that parameters to an instance of a class are stored within the class instance by in-line coding which is a part of the calling sequence.

6.3 Virtual quantities

A virtual quantity V specified in a class Pj in the prefix sequence Pl,.....,Pn will be replaced by the innermost declaration of a matching quantity V of the sequence Pj, ...,Pn. Suppose that this is in Pk. Then the replacement is valid also for Pj,.....,Pk-1. This is implemented by having, in the prototype for the class Pn, one item for each specified virtual quantity in the classes Pl,.....,Pn. These are in fixed positions within the prototype, so that the virtual quantity valid in a specific instance may be found when its index and the class of the B.I. (i.e. its prototype) is known.

In the Common Base, the following specifiers are accepted for virtual quantities:

label, switch, procedure and <type> procedure

A virtual quantity in an instance of a class or a prefixed block will be called "closed" if a declaration matching the specification exists within the object. A virtual which is not closed will be called "open".

It should be noted that a reference to a virtual quantity always must be through the prototype pointer of the actual object.

Subroutines associated with virtual quantities are:

CCVP	Call connected virtual procedure
CDVP	call dot virtual procedure
CVP	call virtual procedure
ENTVIRT	enter virtual procedure
GVL	go to virtual label
CVS	calculate virtual switch

These are discussed in the sections on expressions and labels and switches respectively.