5. PROCEDURE DECLARATION

A procedure deviates from a block in that (1) it has a name and (2) may be referred to at several different places in the program, and (3) that parameters may be given to it when invoked. A procedure shares the property of a block that it is impossible to establish a reference to it or to its interior.

A procedure declaration found in a block heading serves to define the procedure within that block. The declaration is not executed at block entry. A procedure call serves to invoke a procedure, i.e. create data storage and transmit actual parameters. When the end of the procedure body is reached, control returns following the call on the procedure.

In order to refer to the actual parameters supplied to the procedure when invoked, a series of formal parameters is given in the procedure declaration. The actual and formal parameters are matched by their position in the parameter lists when the procedure is invoked.

The formal parameters must be specified. The actual parameter must be compatible with the formal parameter.

A parameter to a procedure may be specified as being one of the six types as well as label, switch, procedure, type procedure and type array.

An additional specification, name or value, may be given. If neither of these are present, a default specification will be assumed dependent upon the mandatory specification mentioned above.

If the declarator procedure is preceded by a type, this indicates that a value of the given type should be associated with the procedure name. Its initial value should be as previously defined for local variables.

If the name of a procedure is used on its own within the procedure body on the left hand side of an assignment or denotes operator, this indicates an operation upon the value associated with the procedure. If it is given on the right hand side of an assignment or denotes operator it will be a recursive call of the procedure unless it is on its own and another assignment or denotes operator is found on its right hand side in the same statement.

As for sub-blocks, it is unnecessary to treat declarations and statements within the body as separate items, as a jump from end of declaration code to statements may be made unconditionally.

The end of the procedure body is indicated by a call on the end procedure (EPR) subroutine.

There are several alternatives open to the implementor for organization of the reference to parameters specified as being called by name.

SIMULA has adopted the scheme from Algol 60 that the location of an actual parameter or subscripted variable on the left hand side of an assignment or denotes operator should be evaluated prior to evaluation of the right hand side.

It would thus be natural to have three subroutines dealing with parameter evaluation: CPL (calculate parameter location), SP (store parameter) and LP (load parameter).

The actual and the formal parameter should be compatible. This means for instance a real variable in a procedure call may correspond to a formal parameter of type integer and vice versa. Several other cases exist which should be properly handled by the runtime system.

```
universal procedure EPR (val);
   begin ref (driver) x; ref (program) out;
     x :- CD.drex;
     out :- CD.pex;
     restore (CD.acs);
     if CD.dot then deletenotice (CD.drp);
     deletenotice (CD);
     CD :- x;
     EPR := val;
     update display;
     go to out;
   end EPR;
```

Comment: "val" is the value to be returned by the procedure from which we are currently returning. Since the type is immaterial to the description of EPR, "val" has not been specified. EPR is called a "universal" procedure, because its value depends on where it was called.