Bibliotheek

1. uitgangspunten.

1.1. Voor real functies van real value argument moet een aanroep bestaan, die het argument in F meekrijgt en het resultaat in F aflevert.

1.2. Bij algemenere parameterlijsten lijkt het wenselijk, dat de parametergegevens overal, maar wel consecutief, kunnen staan, dus op willekeurige plaatsen in het programma, maar ook in de stapel.

1.3. Beide bovenstaande wensen moeten zoveel mogelijk geintegreerd worden.

1.4. Het lijkt aantrekkelijk om Jensens device mogelijk te maken. In het algemeen zou een parameter een expressie moeten kunnen zijn, ook als het een array identifier betreft, of een variabele waaraan een waarde moet worden toegekend.

1.5. Het lijkt geen bezwaar, om als aan een parameter zowel waarden worden toegekend, als eraan waarden worden ontleend, deze parameter twee maal in de parameterlijst te moeten opnemen.

2. uitwerking.

2.1. Op grond van 1.2. moet in een register, b.v. het A-register (dit laat S vrij voor DOS-opdrachten) het beginadres van de parameterlijst kunnen worden meegegeven.

2.2. Op grond van 1.4. moet de bibliotheekroutine via DO(S)-opdrachten beschikken over zijn parameters. In de parameterlijst dienen daarom X8-opdrachten te staan. Voor ingewikkelde parameters mogen dit subroutine-aanroepen zijn. 2.2. impliceert het in 1.5. genoemde.

2.3. Voorbeeld. In de bibliotheek zij een subroutine, die ongeveer ekwivalent is met de procedure:

```
real procedure INPROD (i, n, a, b); integer i, n; real a, b;
begin integer j, k; real s;
      k:= 0; s:= 0; j:= - n;
      for j:= j + 1 step 1 until 0 do
      begin i:= k:= k + 1; s:= s + a × b end ;
      INPROD:= s
end ;
```

Een aanroep, overeenkomend met INPROD (I, 10, VECA[I], VECB[I]), zou er dan in ELAN als volgt kunnen uitzien:

```
        GOTO (:CALL)
PARLI:  I = G           " parameterlijst  i
        F = 10          "                 n
        SUBC (:AA)      "                 a
        SUBC (:BB)      "                 b
AA:     S = :VECA       " subroutine voor VECA[I]
        S + I
        F = MS
        GOTO (MC[-1])
BB:     S = :VECB       " subroutine voor VECB[I]
        GOTO (:AA[1])
I:      'SKIP' 1        " locatie van I
CALL:   A = :PARLI      " adres parameterlist in A meegeven
        SUBC (:INPROD)
```

2.4. In verband met 1.3. zou een subroutine als SQRT er als volgt kunnen uitzien:

```
SQRT:   DO (MA)          " neem argument in F
        ..........       " en lever de wortel in F af
        GOTO (MC[-1])
```

Twee mogelijke aanroepen zijn dan:

```
        F = ARGUMENT
        SUBC (:SQRT[1])
```

en

```
        GOTO (:CALL)
PARLI:  F = ARGUMENT
CALL:   A = :PARLI
        SUBC (:SQRT)
```

Het zou derhalve mogelijk moeten zijn, om bibliotheekroutines verschillende ingangen te geven. Dit lijkt me ueberhaupt een desideratum.

2.5. Ingewikkelder voorbeeld. Een integratieroutine analoog aan real procedure INT (x, a, b, Fx, eps) zou in ELAN volgens het bovenstaande als volgt kunnen worden aangeroepen:

```
        GOTO (:CALL)
PARLI:  X = F            " parameterlijst van INT: x
        F = ONDERGRENS   " a
        F = BOVENGRENS   " b
        SUBC (:FX)       " Fx
        F = EPS          " eps
FX:     A = :PL
        GOTO (:FCT)      " bereken Fx (x, .5)
PL:     F = X            " eerste parameter van Fx
        F = .5           " tweede parameter van Fx
X:      'SKIP' 2         " locatie van X
CALL:   A = :PARLI
        SUBC (:INT)
```

Het zal duidelijk zijn, dat zoiets met zorg opgeschreven zal moeten worden. De moeilijkheid komt voort uit het feit, dat aan Fx weer een parameterlijst moet worden meegegeven. Het in 3. geschetste alternatief van 2. is als geheel ingewikkelder, maar geeft mogelijk een machtiger apparaat voor het aan elkaar knopen van subroutines.

3. ingewikkelder alternatief.

3.1. Iedere routine heeft een ingang, waarbij niet in A het beginadres van de parameterlijst wordt meegegeven, maar in S een adres, zodanig dat de opdracht DO (MS[1]) in A het beginadres van de parameterlijst aflevert. Verder beslaat iedere parameter in de lijst twee plaatsen, een X8 opdracht, en verdere informatie. De routines beschikken over hun parameters via DOS-opdrachten. In dit geval zou bovenstaande aanroep van INT er nu als volgt uitzien:

```
        S = :CALL
        GOTO (:CALL)
PARLI:  X = F           " parameterlijst van INT
        + 0
        F = ONDERGRENS
        + 0
        F = BOVENGRENS
        + 0
        SUBC (:FCT)
        A = :PL
        F = EPS
        + 0
PL:     F = X           " parameterlijst voor FCT
        + 0
        F = .5
        + 0
X:      'SKIP' 2        " locatie van X
CALL:   SUBC (:INT)
        A = :PARLI
```

Ook hier zou men weer kunnen denken aan verschillende ingangangen, al naar gelang de ingewikkeldheid van de parametersituatie.