Strukturerad Programmering

Niklaus Wirth, 1976:

Algorithms + Datastructures = Programs

```
C     A weird program for calculating Pi written in Fortran.
C     From: Fink, D.G., Computers and the Human Mind, Anchor Books, 1966.

      PROGRAM PI
      DIMENSION TERM(100)
      N=1
    3 TERM(N)=((-1)**(N+1))*(4./(2.*N-1.))
      N=N+1
      IF (N-101) 3,6,6
    6 N=1
    7 SUM98 = SUM98+TERM(N)
      WRITE(*,28) N, TERM(N)
      N=N+1
      IF (N-99) 7, 11, 11
   11 SUM99=SUM98+TERM(N)
      SUM100=SUM99+TERM(N+1)
      IF (SUM98-3.141592) 14,23,23
   14 IF (SUM99-3.141592) 23,23,15
   15 IF (SUM100-3.141592) 16,23,23
   16 AV89=(SUM98+SUM99)/2.
      AV90=(SUM99+SUM100)/2.
      COMANS=(AV89+AV90)/2.
      IF (COMANS-3.1415920) 21,19,19
   19 IF (COMANS-3.1415930) 20,21,21
   20 WRITE(*,26)
      GO TO 22
   21 WRITE(*,27) COMANS
   22 STOP
   23 WRITE(*,25)
      GO TO 22
   25 FORMAT('ERROR IN MAGNITUDE OF SUM')
   26 FORMAT('PROBLEM SOLVED')
   27 FORMAT('PROBLEM UNSOLVED', F14.6)
   28 FORMAT(I3, F14.6)
      END
```

"Spaghetti code"

In 1968 Dijkstra published a letter, "Go To statement considered harmful" in which he condemned the indiscriminate use of the goto statement [1]. One statement stands out:

"The go to statement as it stands is just too primitive, it is too much an invitation to make a mess of one's program".

Instead he advocated the use of procedures, and conditional and repetition for program control.

[https://craftofcoding.files.wordpress.com/2013/10/lore_spaghetti.pdf]

Spaghetti code forts.

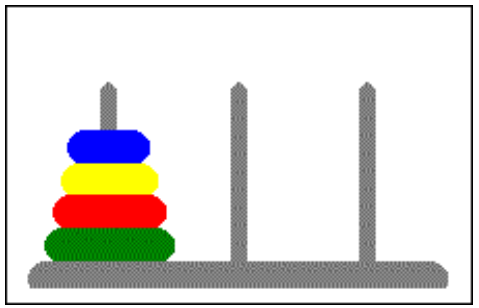
```
10 i = 0
20 i = i + 1
30 PRINT i; " squared = "; i * i
40 IF i >= 10 THEN GOTO 60
50 GOTO 20
60 PRINT "Program Completed."
70 END
```

Here is the same code written in a structured programming style:

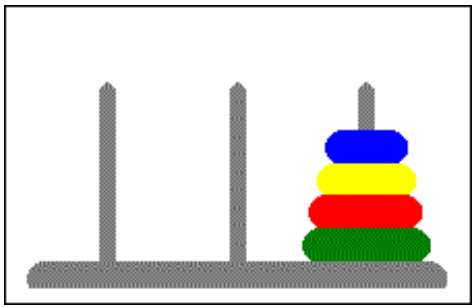
```
10 FOR i = 1 TO 10
20     PRINT i; " squared = "; i * i
30 NEXT i
40 PRINT "Program Completed."
50 END
```

https://en.wikipedia.org/wiki/Spaghetti_code

Programspråket kan hjälpa till - eller inte



Towers of Hanoi



I sed:

```
                s~^xx*$~:n:3:2:1:&:~;tB;d;:B;/^:$/d;h
                s~^:.:\(.\):.:\(.\):*:.*~\2 --> \1~;x
                /^:y:.:.:.:*:.*/b0;/^:n:.:.:.:x:.*/b1
s~:n:\(.\):\(.\):\(.:x*\)x:\(.*\)~:n:\2:\1:\3:y:\1:\2:\3x:\4~
      bB;:1;x;p;x;s~^:n:.:.:.:x:\(.*\)~:\1~;bB;:0;x;p;x
      s~^:y:\(.\):\(.\):\(.\):x\(x*:*\)~:n:\1:\3:\2:\4~
                                bB
```

En flygsimulator - i C

```
#include                                    <math.h>
#include                                   <sys/time.h>
#include                                   <X11/Xlib.h>
#include                                   <X11/keysym.h>
                                           double L ,o ,P
                                          ,_=dt,T,Z,D=1,d,
                                          s[999],E,h= 8,I,
                                          J,K,w[999],M,m,O
                                          ,n[999],j=33e-3,i=
                                          1E3,r,t, u,v ,W,S=
                                          74.5,l=221,X=7.26,
                                          a,B,A=32.2,c, F,H;
                                          int N,q, C, y,p,U;
                                         Window z; char f[52]
                                       ; GC k; main(){ Display*e=
 XOpenDisplay( 0); z=RootWindow(e,0); for (XSetForeground(e,k=XCreateGC (e,z,0,0),BlackPixel(e,0))
; scanf("%lf%lf%lf",y +n,w+y, y+s)+1; y ++); XSelectInput(e,z= XCreateSimpleWindow(e,z,0,0,400,400,
0,0,WhitePixel(e,0) ),KeyPressMask); for(XMapWindow(e,z); ; T=sin(O)){ struct timeval G={ 0,dt*1e6}
; K= cos(j); N=1e4; M+= H*_; Z=D*K; F+=_*P; r=E*K; W=cos( O); m=K*W; H=K*T; O+=D*_*F/ K+d/K*E*_; B=
sin(j); a=B*T*D-E*W; XClearWindow(e,z); t=T*E+ D*B*W; j+=d*_*D-_*F*E; P=W*E*B-T*D; for (o+=(I=D*W+E
*T*B,E*d/K *B+v+B/K*F*D)*_; p<y; ){ T=p[s]+i; E=c-p[w]; D=n[p]-L; K=D*m-B*T-H*E; if(p [n]+w[ p]+p[s
]== 0|K <fabs(W=T*r-I*E +D*P) |fabs(D=t *D+Z *T-a *E)> K)N=1e4; else{ q=W/K *4E2+2e2; C= 2E2+4e2/ K
 *D; N-1E4&& XDrawLine(e ,z,k,N ,U,q,C); N=q; U=C; } ++p; } L+=_* (X*t +P*M+m*l); T=X*X+ l*l+M *M;
  XDrawString(e,z,k ,20,380,f,17); D=v/l*15; i+=(B *l-M*r -X*Z)*_; for(; XPending(e); u *=CS!=N){
                                       XEvent z; XNextEvent(e ,&z);
                                           ++*((N=XLookupKeysym
                                             (&z.xkey,0))-IT?
                                             N-LT? UP-N?& E:&
                                             J:& u: &h); --*(
                                             DN -N? N-DT ?N==
                                             RT?&u: & W:&h:&J
                                              ); } m=15*F/l;
                                              c+=(I=M/ l,l*H
                                              +I*M+a*X)*_; H
                                              =A*r+v*X-F*l+(
                                              E=.1+X*4.9/l,t
                                              =T*m/32-I*T/24
                                               )/S; K=F*M+(
                                               h* 1e4/l-(T+
                                               E*5*T*E)/3e2
                                               )/S-X*d-B*A;
                                               a=2.63 /l*d;
                                               X+=( d*l-T/S
                                                *(.19*E +a
                                                *.64+J/1e3
                                                )-M* v +A*
                                                Z)*_; l +=
                                                K *_; W=d;
                                                sprintf(f,
                                                "%5d  %3d"
                                                "%7d",p =l
                                               /1.7,(C=9E3+
                                    O*57.3)%0550,(int)i); d+=T*(.45-14/l*
                                   X-a*130-J* .14)*_/125e2+F*_*v; P=(T*(47
                                   *I-m* 52+E*94 *D-t*.38+u*.21*E) /1e2+W*
                                   179*v)/2312; select(p=0,0,0,0,&G); v-=(
                                    W*F-T*(.63*m-I*.086+m*E*19-D*25-.11*u
                                     )/107e2)*_; D=cos(o); E=sin(o); } }
```

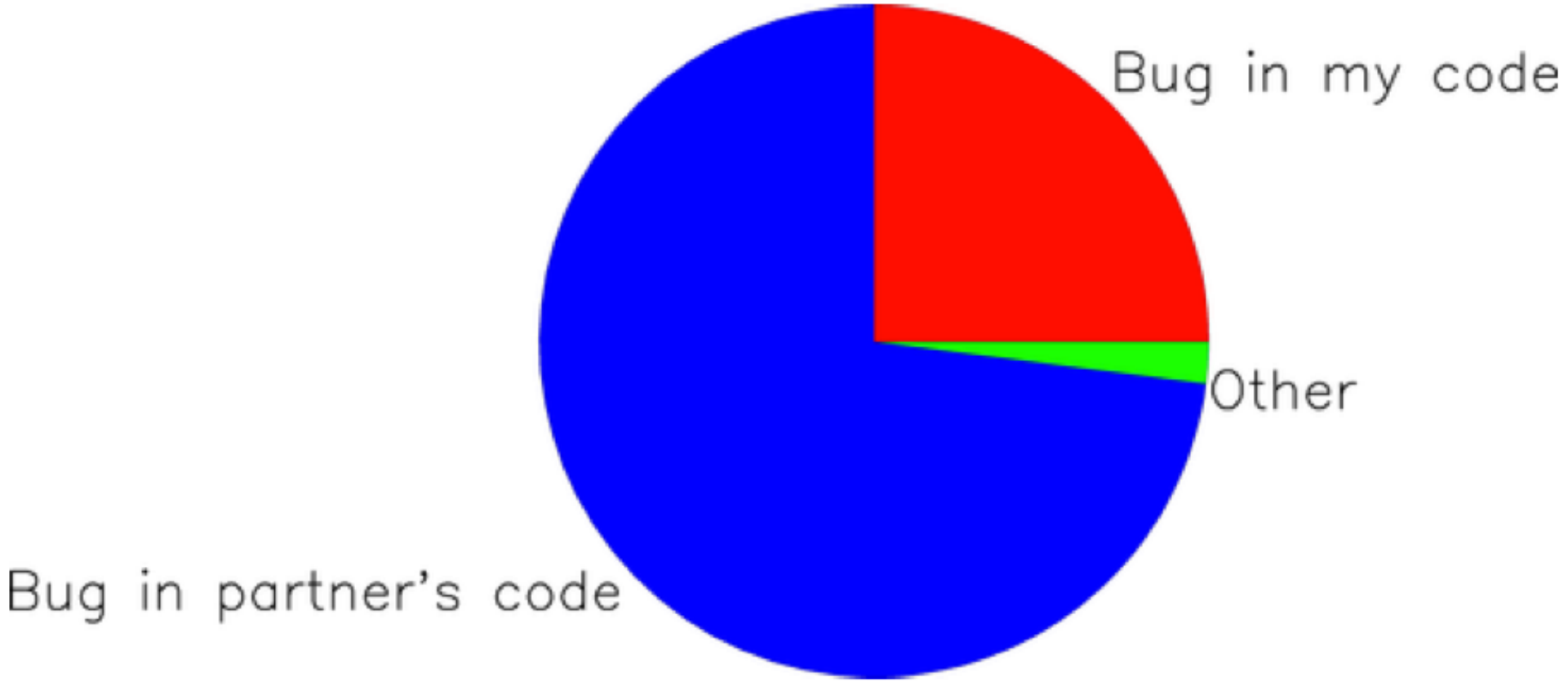
RC4-krypto - i FORTH

```
0 value ii          0 value jj
0 value KeyAddr     0 value KeyLen
create SArray     256 allot     \ state array of 256 bytes
: KeyArray        KeyLen mod    KeyAddr ;

: get_byte        + c@ ;
: set_byte        + c! ;
: as_byte         255 and ;
: reset_ij        0 TO ii    0 TO jj ;
: i_update        1 +    as_byte TO ii ;
: j_update        ii SArray get_byte +    as_byte TO jj ;
: swap_s_ij
    jj SArray get_byte
       ii SArray get_byte  jj SArray set_byte
    ii SArray set_byte
;

: rc4_init ( KeyAddr KeyLen -- )
    256 min TO KeyLen    TO KeyAddr
    256 0 DO    i i SArray set_byte    LOOP
    reset_ij
    BEGIN
        ii KeyArray get_byte    jj +  j_update
        swap_s_ij
        ii 255 < WHILE
        ii i_update
    REPEAT
    reset_ij
;
: rc4_byte
    ii i_update    jj j_update
    swap_s_ij
    ii SArray get_byte    jj SArray get_byte +    as_byte SArray get_byte  xor
;
```

Vem programmerar fel?

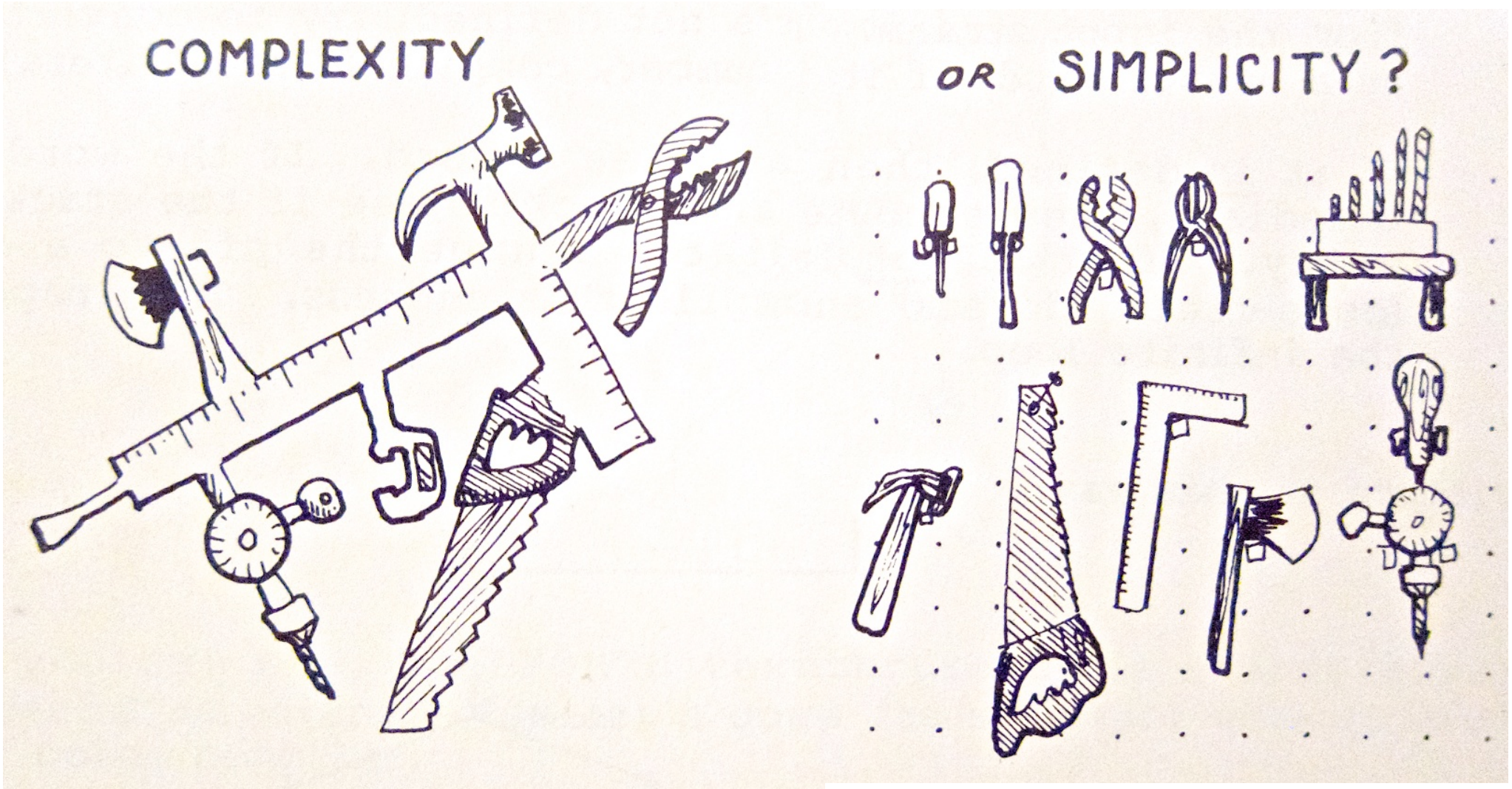


http://blog.regehr.org/archives/199

Vem programmerar fel?

http://blog.regehr.org/archives/199

Vi har ett val...



Från: Leo Brodie, Starting FORTH

JSP
-
Jackson Structured Programming

Här förenklad, "light"-version

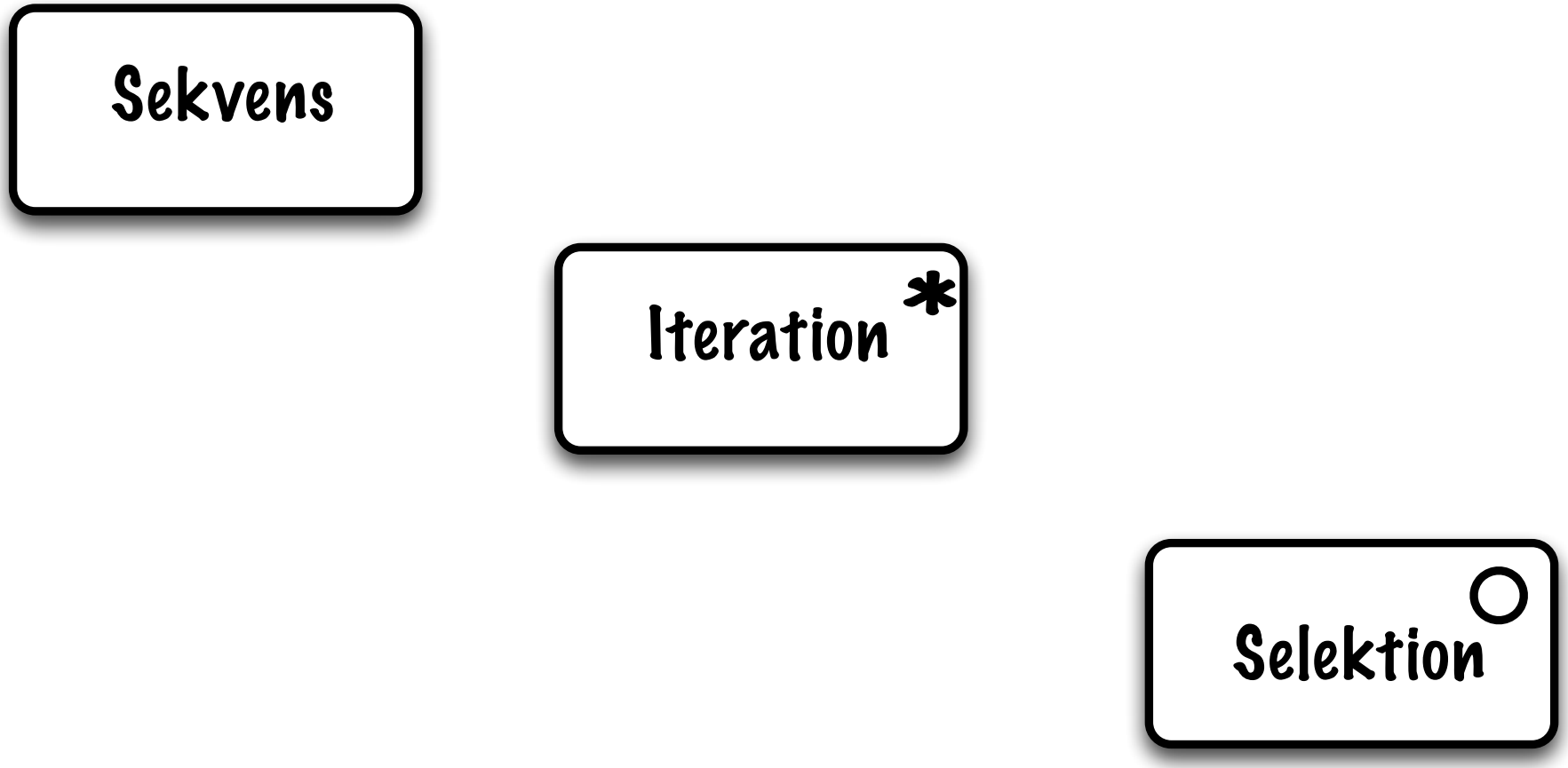
JSP-princip:

Data kan beskrivas i strukturdiagram.

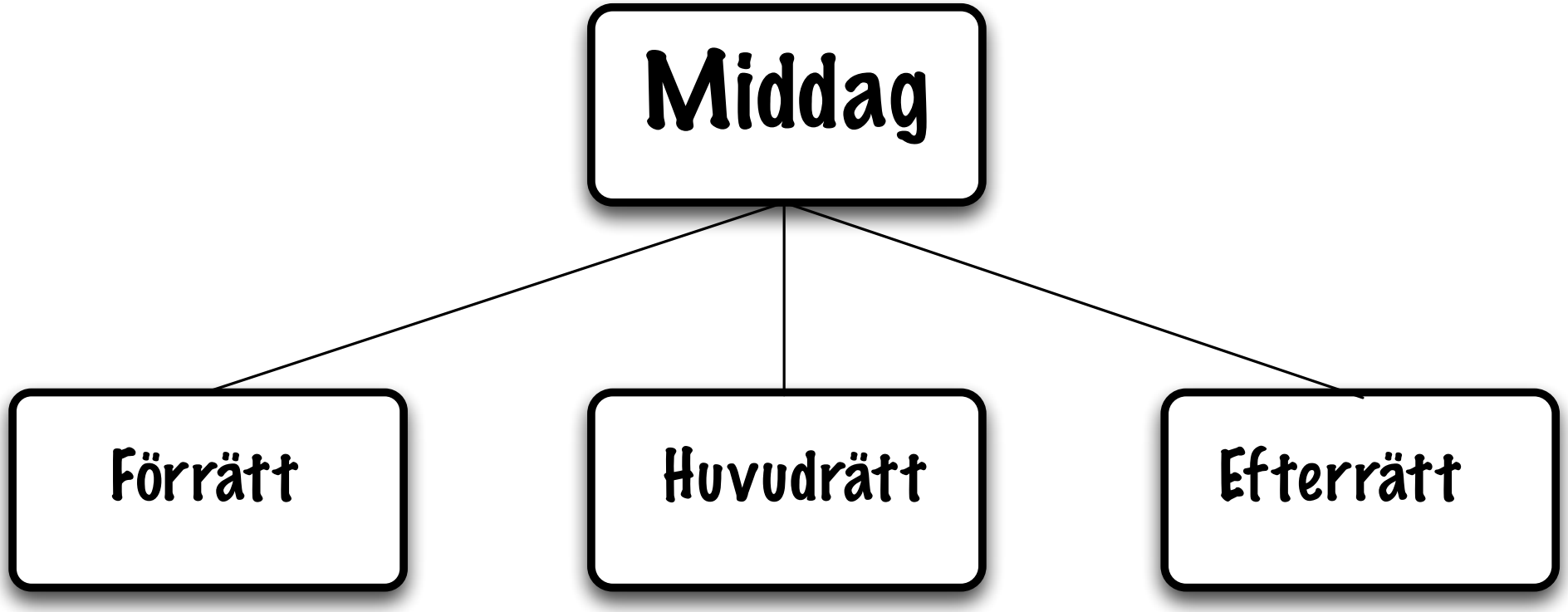
Programmet skrivs i samklang med datat.

Algorithms + Datastructures = Programs

Strukturdiagram Byggblock



Sekvens: först det ena, sen det andra och sen...

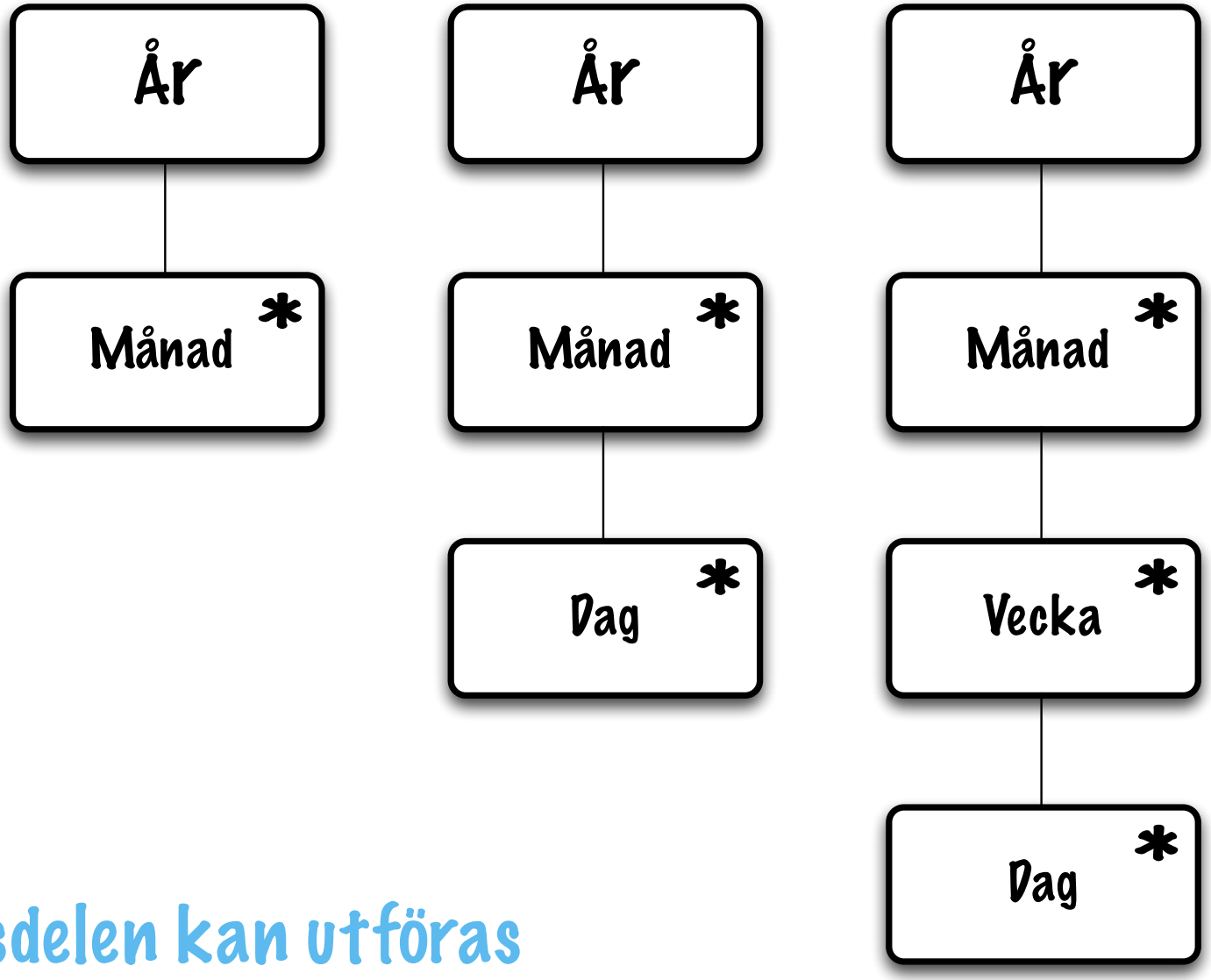


- Initiera hårdvara
- Definiera konstanter
- ...

- Utför databehandlingen
- Leverera delresultat
- ...

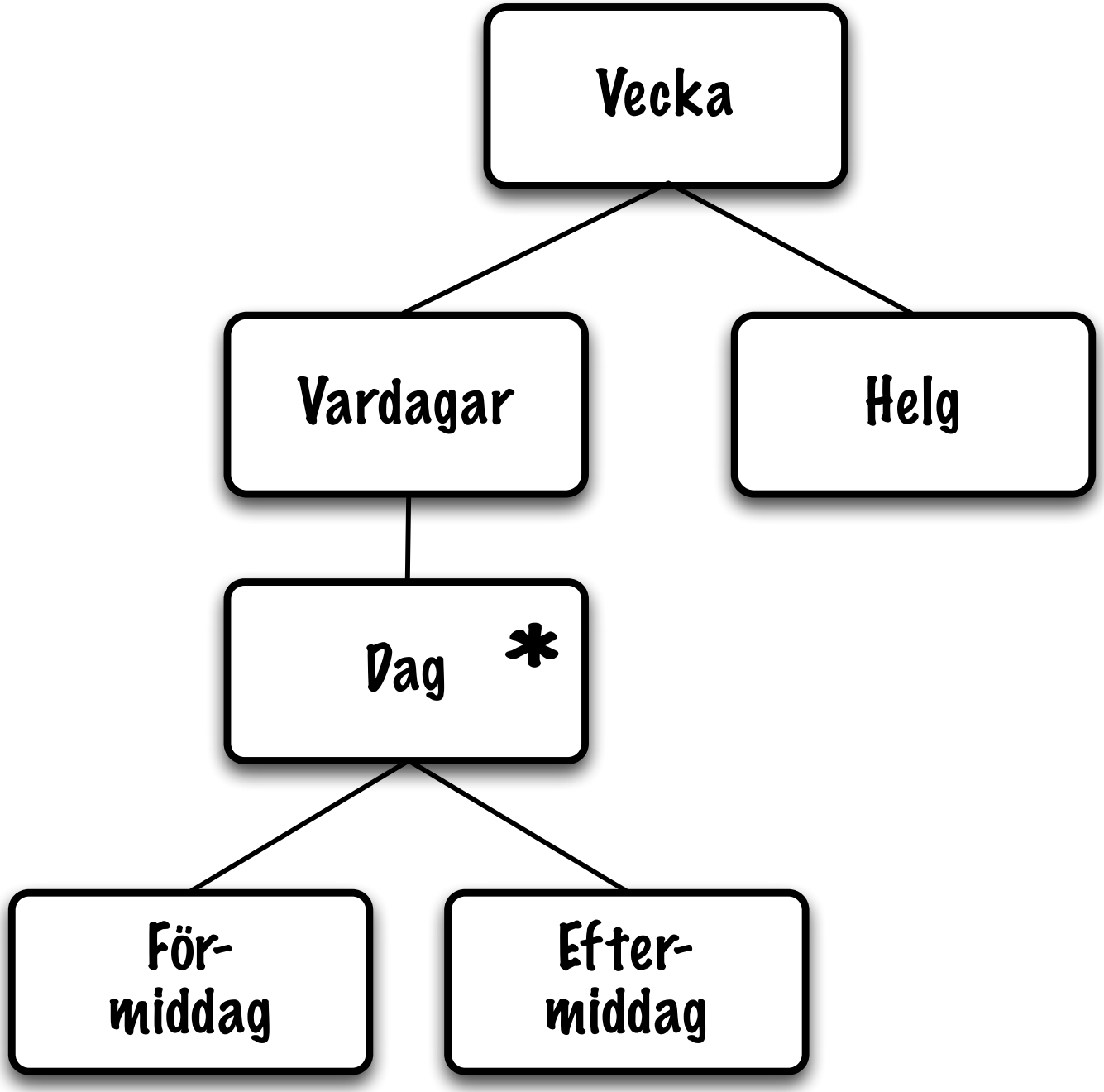
- Slutresultat
- Avsluta HW
- ...

Iteration: om och om igen och igen...

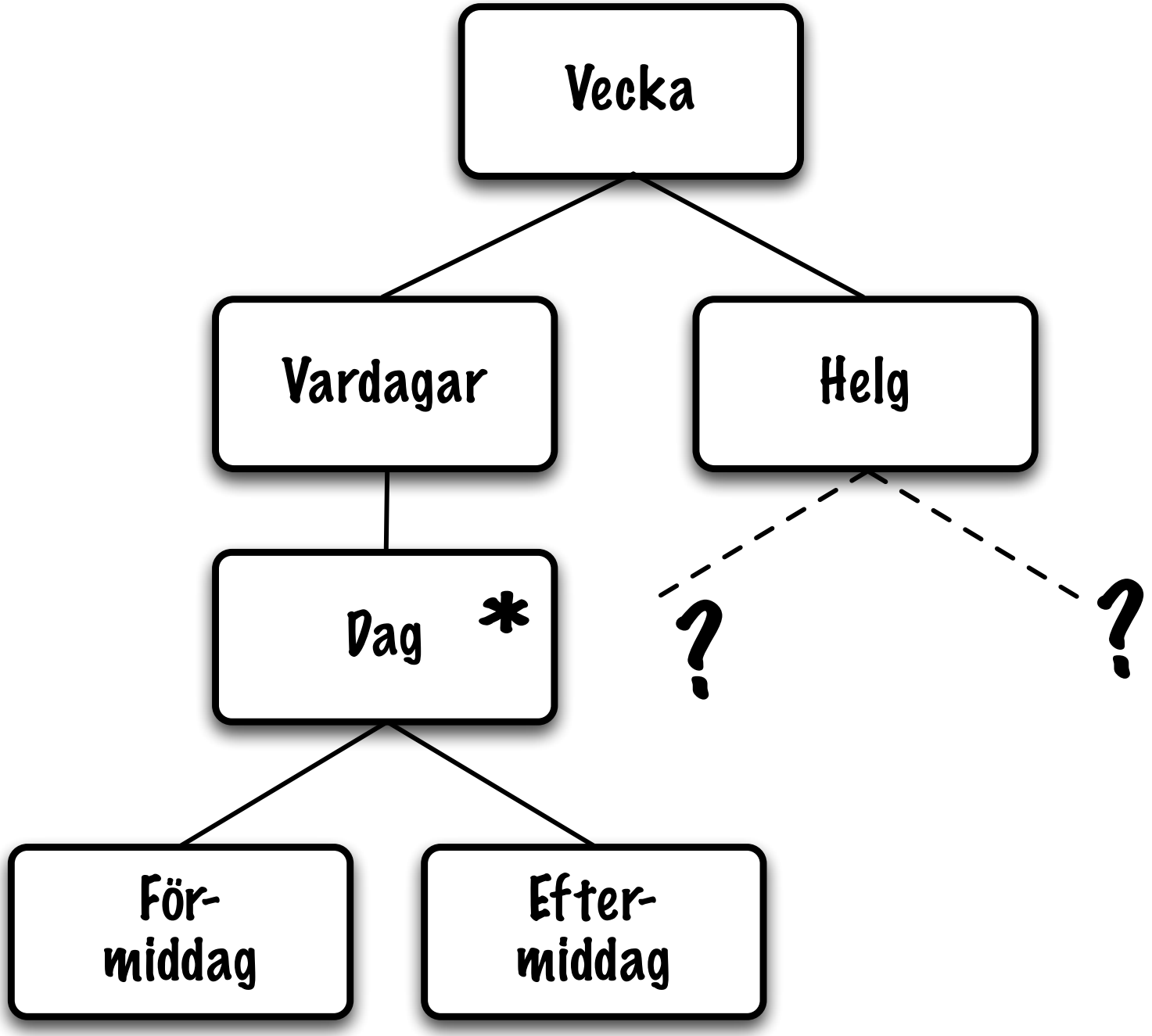


Iterationsdelen kan utföras
NOLL gånger!

Kombinerad sekvens och iteration

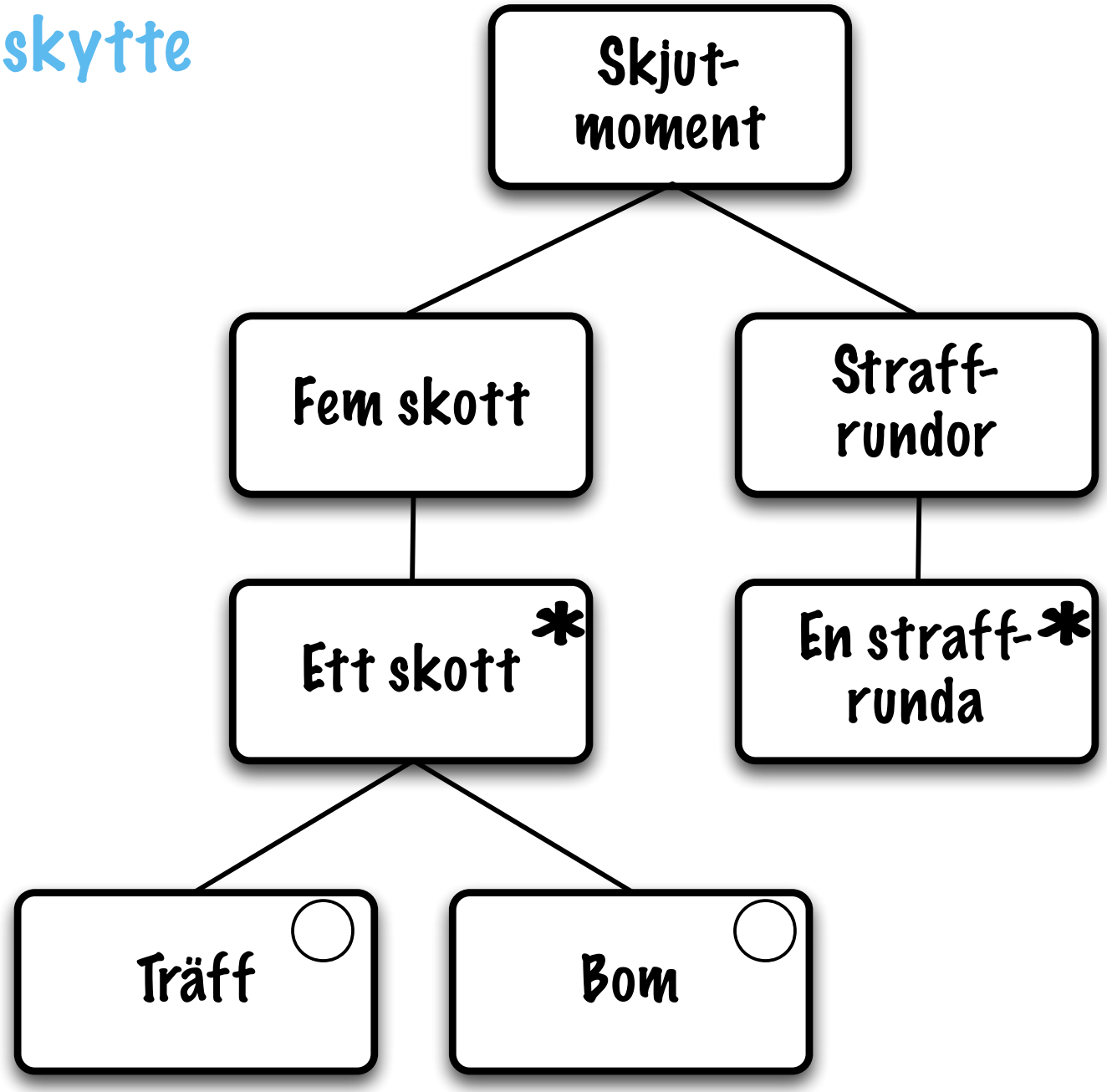


Kombinerad sekvens och iteration. Forts.



Selektion: det ena eller det andra eller ...

Skidskytte



JSP-Regler

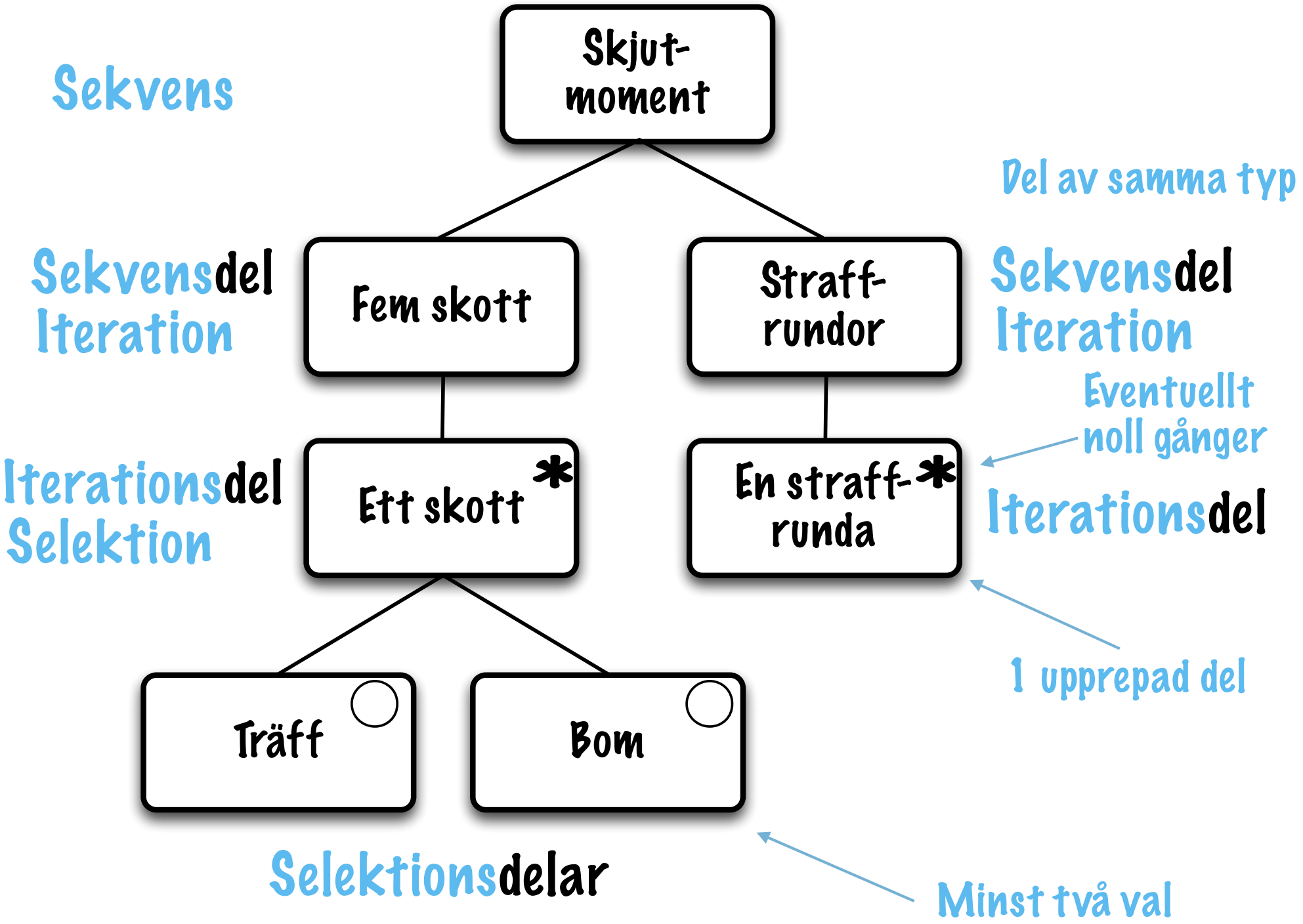
En komponent får endast ha delar av samma typ:
- En sekvens får inte bestå av iterationsdelar.
- En selektion får inte bestå av sekvensdelar osv.

En iteration får bara ha en (1) iterationsdel.

En iteration kan ske noll gånger:
- Tänk `while()` inte `if()`.

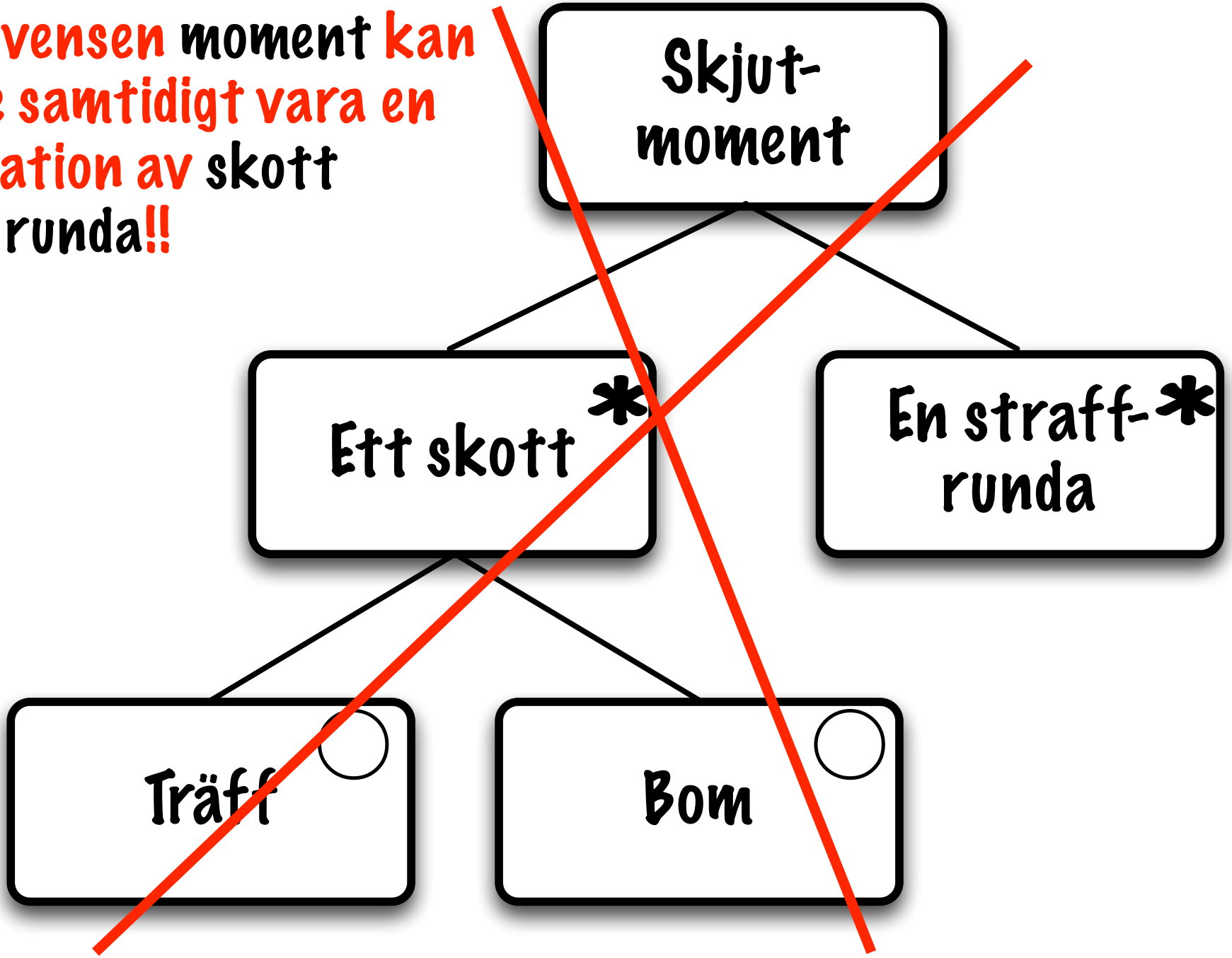
En selektion måste ha minst två delar:
- Ett val måste ha mer än ett alternativ.
- Ofta ett defaultalternativ också.

Korrekt enligt JSP

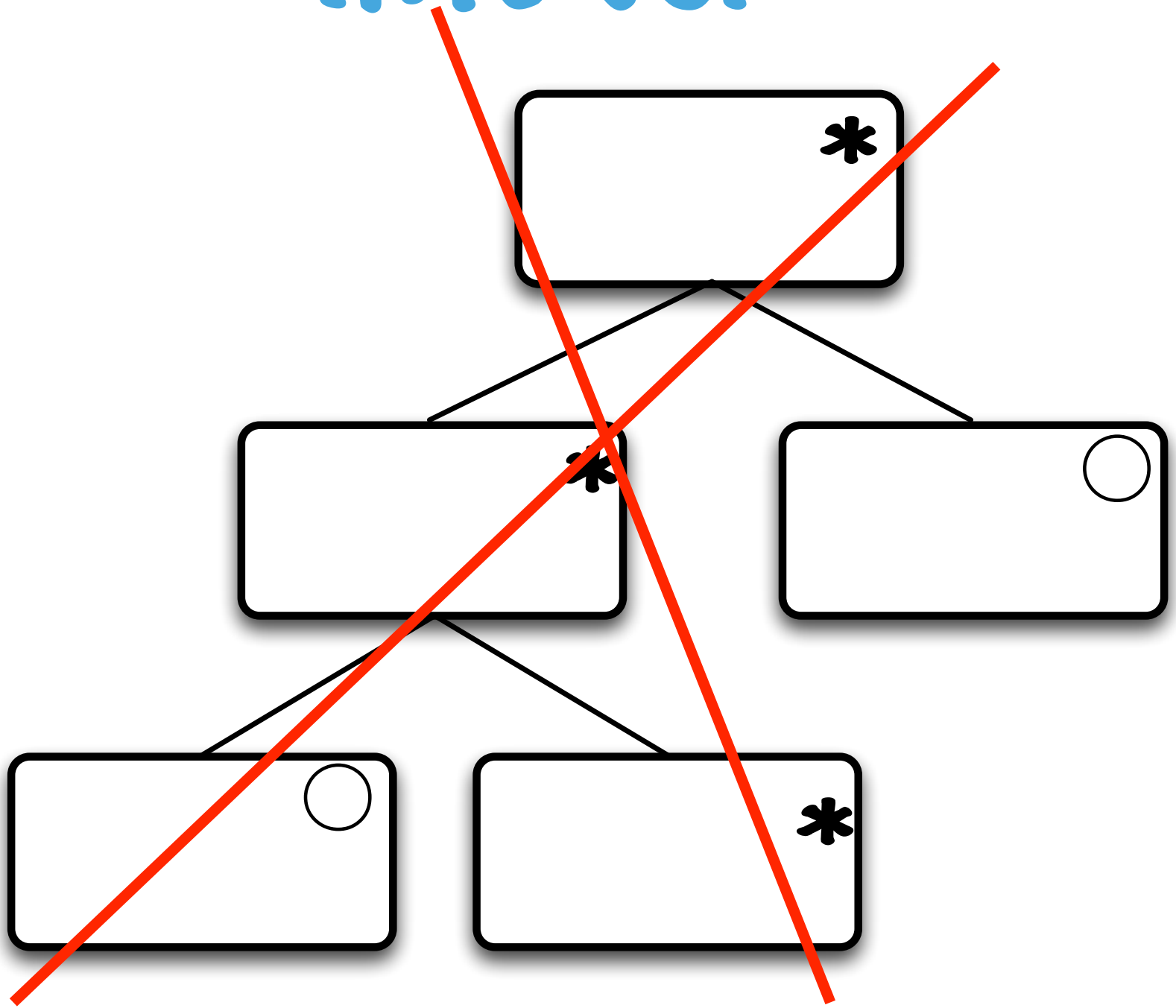


Inte korrekt enligt JSP

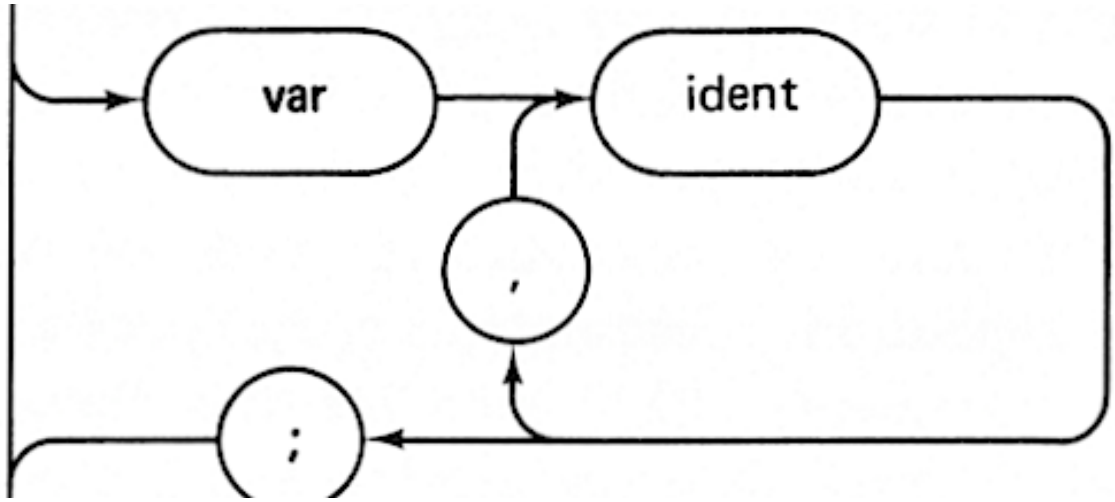
Sekvensen moment kan inte samtidigt vara en iteration av skott och runda!!



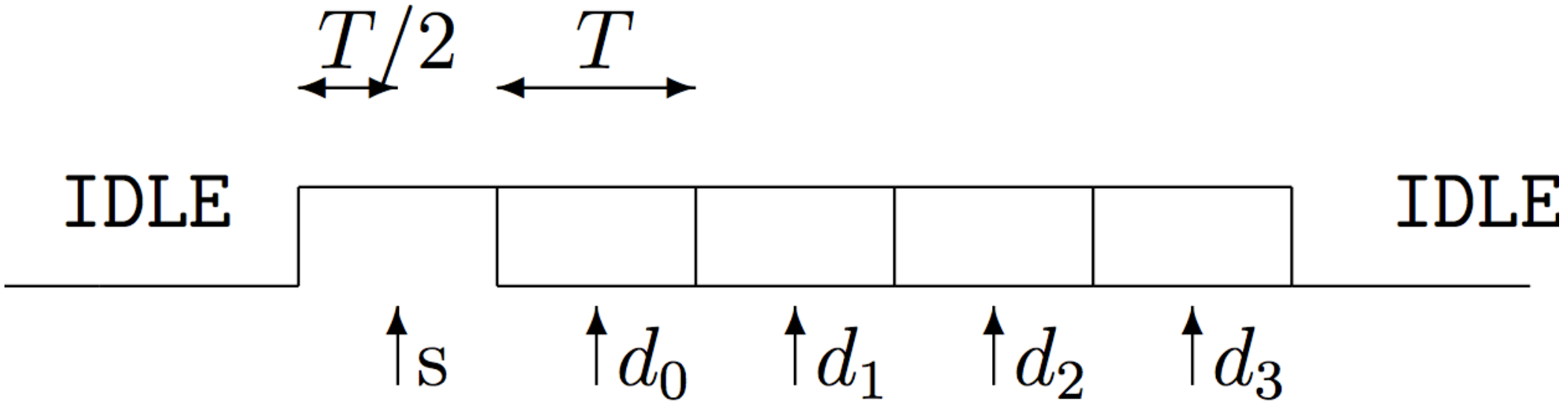
Inte JSP



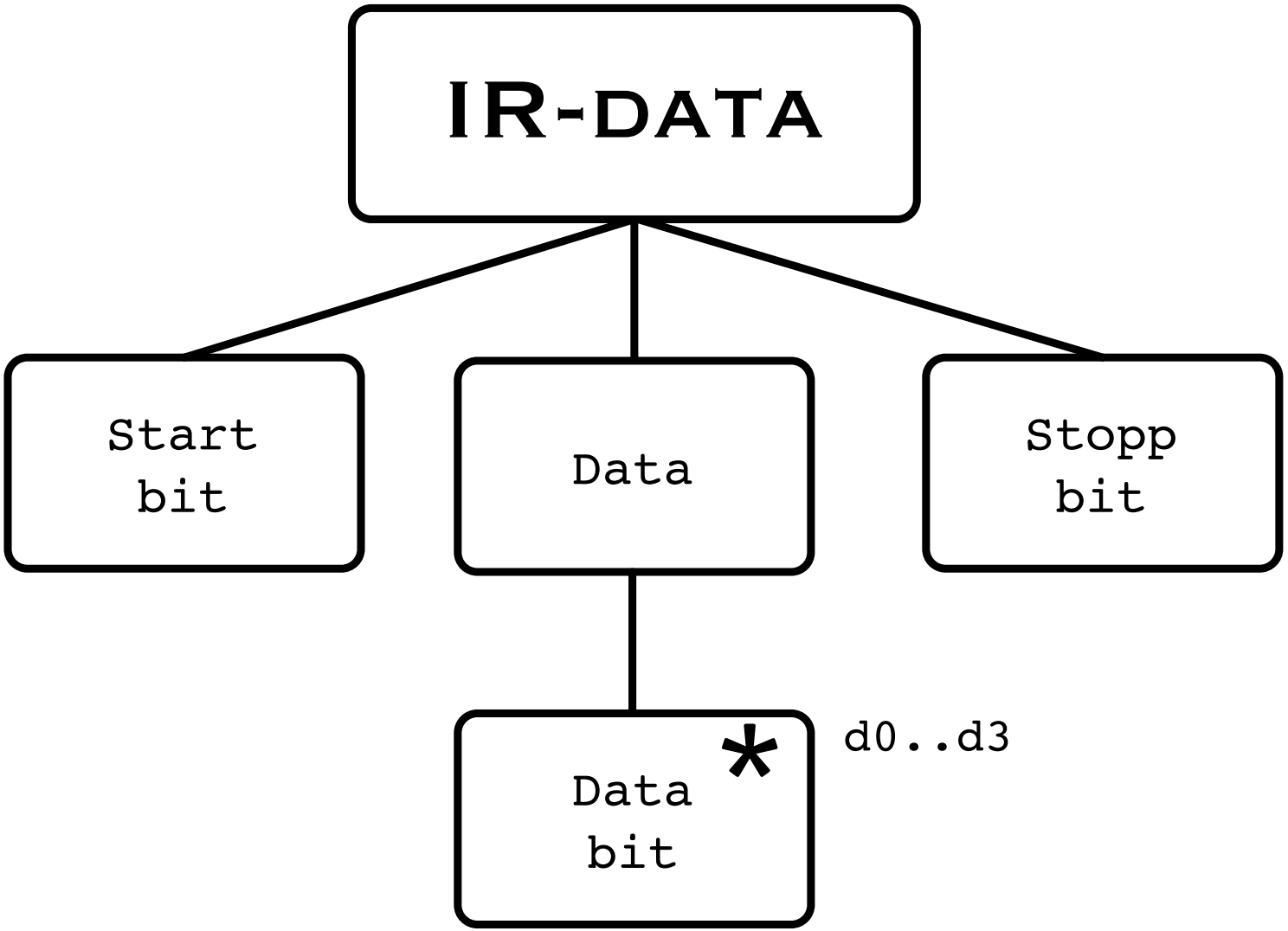
Gå från en beskrivning till en annan



Labl. Seriell signal fyra bitar



Strukturdiagram IR-signal



Övergång till kod

Sekvens

```
    :
   call A
   call B
   call C
    :
```

Iteration

```
      :
AGAIN:
     cpi  r16,N
     breq DONE
     call B
     call C
     jmp  AGAIN
DONE:
      :
```

Selektion

```
      :
     cpi  r16,N
     brne A1
     call R
     jmp  DONE
A1:
     cpi  r16,M
     brne A2
     call S
     jmp  DONE
A2:
      :
DONE:
      :
```

Fler övningsuppgifter i häftet.

Laborationerna beskrivs i form av JSP.