

Amendment Design for Bean Plant Care

1. Introduction

This document is an amendment of the technical report 'Bean Plant Care Design' number TR/CLAES/91/99.11. This amendment takes into account the comments mentioned in the work shop of Bean and enhancement in the interface and task. It contains also some modifications in the domain knowledge.

2. Domain Knowledge

2.1 Domain Ontology

2.1.1 New Concepts

The following concepts are added.

concept user request;

properties:

value: { }

source of value: user;
cardinality: single;

concept operation cost;

properties:

value: { , };

source of value: user ;
cardinality: single;

concept session date;

properties:

current month: numeric;
source of value: system;
cardinality: single;

value: date; /* system date */
source of value: system;
cardinality: single;

concept Irrigation;
properties:
type: { };
source of value: database;
cardinality: single;

concept Planting;
properties:
type: { };
source of value: database;
cardinality: single;

concept ;
sub-type-of: operation;
importance: ;
tool name:
method type: mechanical
total time: 4

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 4

concept ;
sub-type-of: operation;
importance: ;
tool name: { , }
source of value: database;
cardinality: single;
method type: mechanical;
total time: 4

concept ;
sub-type-of: operation;
importance: ;
tool name: +
method type: mechanical
total time: 2

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 4

concept ;
sub-type-of: operation;
importance: ;
tool name:
method type: mechanical
total time: 4

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 6

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 4

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 8

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 2

concept ;
sub-type-of: operation;
importance: ;
application date: date
source of value: derived(SUGGEST);
cardinality: single;
method type: manual
labor age: men
number of men per feddan: 4

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 6

2.1.2 Updated Concepts

The following concepts are updated. The underline and large font items are added. The italic and large items are deleted.

concept plant;
properties:

status: }

source of value: user;
cardinality: single;

possible status: }

source of value: derived;
cardinality: multiple ;

age: numeric;
source of value: derived;
cardinality: single;

variety type: { , };
source of value: derived;
cardinality: single;

concept plantation;

properties:

type: { }
source of value: data base;
cardinality: single;

status: { }
source of value: user;
cardinality: multiple;

possible status: { }
source of value: user;
cardinality: multiple;

plant density: { }
source of value: user;
cardinality: single;

appearance: { }
source of value: user;
cardinality: single;

crop type: { }
source of value: data base;
cardinality: single;

date: universal ;
source of value: data base;
cardinality: single;

area: numeric % in feddan
source of value: database;
cardinality: single;

concept operation;

properties:

status: {suggested, not suggested, cancelled};
source of value: derived;
cardinality: single;

number of boys per feddan: numeric;
source of value: derived ;
cardinality: single;

number of men per feddan: numeric;
source of value: derived ;
cardinality: single;

operation number: numeric;
source of value: derived(APPLY);
cardinality: single;

COST: numeric;
source of value: derived;
cardinality: single;
default: 0

actual cost: numeric;
source of value: user;
cardinality: single;
default: 0

name: universal;
source of value: derived(APPLY);
cardinality: multiple;

concept ;
sub-type-of: operation;
importance ;
method type: manual
labor age: men
number of men per feddan: 1

concept ;
sub-type-of: operation;
importance ;
tool name:
method type: mechanical
total time: 4

concept ;
sub-type-of: operation;
importance: ;
tool name: +
method type: mechanical
total time: 4

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 3

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 4

concept ;
sub-type-of: operation;
importance: ;
tool name:
method type: mechanical
total time: 4

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 6

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 4

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 2

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 1

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: men
number of men per feddan: 1

concept ;
sub-type-of: operation;
importance: ;
method type: manual
labor age: boys
number of boys per feddan: 60

concept ;
sub-type-of: operation;
importance: ;
tool name:
method type: mechanical
total time: 6

2.1.3 Deleted Concepts

The following concepts are deleted.

concept ;
sub-type-of: operation;
importance: ;

concept ;
sub-type-of: operation;
importance: ;

concept ;
sub-type-of: operation;
importance: ;

concept ;
sub-type-of: operation;
importance: ;

concept ;
sub-type-of: operation;
importance: ;

concept ;
sub-type-of: operation;
importance: ;

concept ;
sub-type-of: operation;
importance: ;

concept ;
sub-type-of: operation;
importance: ;

2.1.4 New Relation

The following relation is added.

relation: APPLY;
argument-1: plantation, plant, irrigation, soil;
argument-role: environment;
argument-2: operation;
argument-role: agricultural operations;

2.2 Domain Model

domain-model: suggestion model;
parts: tuple(suggest);

The modifications of this model is huge, so the following is the complete suggestion model

axioms:

The following rules are added:

```

                : occurrence =                &
Plantation: type =                &
(session date : current month = 10 OR
 session date : current month = 11)
    SUGGEST
(                : status = suggested )

                : occurrence =                &
Plantation: type =                &
(session date : current month = 2 OR
 session date: current month = 3 OR
 session date : current month = 8 OR
 session date : current month = 9 OR
)
    SUGGEST
(                : status = suggested )

                : occurrence =                &

```

Plantation: type = &
 (session date : current month = 10 OR
 session date : current month = 11 OR
 session date : current month = 12
)
 SUGGEST
 (: status = suggested)

 (: occurrence = &
 : occurrence = &

 soil: type = &
 (session date : current month = 10 OR
 session date : current month = 9) &
 session date : value - : application date >= 2)
 SUGGEST
 (: status = suggested)

 (: occurrence = &
 : occurrence = &

 soil: type = &
 session date : current month = 8 &
 session date : value - : application date >= 1)
 SUGGEST
 (: status = suggested)

 (: occurrence = &
 : occurrence = &

 soil: type = &
 (session date : current month = 2 OR
 session date : current month = 3 OR
 session date : current month = 11 OR
 session date : current month = 12) &
 session date : value - : application date >= 3)
 SUGGEST
 (: status = suggested)

 (: occurrence = &
 : occurrence = &

soil: type = &
(session date : current month = 10 OR
session date : current month = 9) &
session date : value - : application date >= 6)
SUGGEST
(: status = suggested)

: occurrence = &

soil: type = &
session date : current month = 8 &
session date : value - : application date >= 4)
SUGGEST
(: status = suggested)

: occurrence = &

soil: type = &
(session date : current month = 2 OR
session date : current month = 3 OR
session date : current month = 11 OR
session date : current month = 12) &
session date : value - : application date >= 8)
SUGGEST
(: status = suggested)

: occurrence = &

soil: type = &
(session date : current month = 10 OR
session date : current month = 9) &
session date : value - : application date >= 10)
SUGGEST
(: status = suggested)

: occurrence = &

soil: type = &
session date : current month = 8 &
session date : value - : application date >= 8)
SUGGEST
(: status = suggested)

```

: occurrence = &
soil: type = &
(session date : current month = 2 OR
session date : current month = 3 OR
session date : current month = 11 OR
session date : current month = 12) &
session date : value - : application date >= 12)
SUGGEST
( : status = suggested )

( : occurrence = &
: occurrence = &
(soil: type = OR
previous crop : fertilizer = ))
SUGGEST
( : status = suggested )

: occurrence = &
((soil: type = OR : type = ) &
previous crop : fertilizer = ))
SUGGEST
( : status = cancelled)

: occurrence = &
( : occurrence = OR
( : occurrence = &
: occurrence = ))
SUGGEST
( : status = suggested )

: occurrence = &
( : status = cancelled &
: occurrence = )
SUGGEST
( : status = suggested )

: occurrence = &

```

: occurrence = &
session date : value - : application date >= 5
SUGGEST
(: status = suggested)

: occurrence = &
Irrigation: type = &
: occurrence = &
session date : value - : application date >= 1
SUGGEST
(: status = suggested)

: occurrence = &
Irrigation: type = &
: occurrence =
SUGGEST
(: status = cancelled)

: occurrence = &
(: occurrence = OR
: occurrence = OR
: status = cancelled)
SUGGEST
(: status = suggested)

: occurrence = &
Irrigation: type = &
: occurrence =
SUGGEST
(: status = suggested)

: occurrence = &
Irrigation: type = &
: occurrence =
SUGGEST
(: status = suggested)

```

: occurrence = &
Irrigation: type = &
: occurrence =
SUGGEST
( : status = suggested )

: occurrence = &
: occurrence = &
Irrigation: type =
SUGGEST
( : status = suggested )

: occurrence = &
Irrigation: type = &
: occurrence =
SUGGEST
( : status = suggested )

: occurrence = &
Irrigation: type = &
(soil: type = OR
soil: type = ) &
: occurrence =
SUGGEST
( : status = suggested )

: occurrence = &
Soil: type = &
: occurrence =
SUGGEST
( : status = suggested )

: occurrence = &
: occurrence = &
Soil: type = &
session date : value - : application date >= 6
SUGGEST

```


(: status = suggested)

: occurrence = &

: occurrence = &

Soil: type = &

session date : value - : application date >= 10

SUGGEST

(: status = suggested)

: occurrence = &

: occurrence = &

SUGGEST

(: status = suggested)

: occurrence = &

: occurrence = &

Plantation: type =

SUGGEST

(: status = suggested)

Irrigation: type = &

Planting: type = &

: occurrence = &

: occurrence = &

not(: occurrence =) &

not(: occurrence =)

SUGGEST

(: status = suggested)

: occurrence = &

plant: age > 10 &

plant: age < 30 &

plant: status = &

Plantation: status =

SUGGEST

(: status = suggested)

```

: occurrence = &
plantation : status = &
: occurrence =
SUGGEST
( : status = suggested )

: occurrence = &
not(plantation : status = )&
: occurrence =
SUGGEST
( : status = cancelled)

: occurrence = &
plantation : status = &
: occurrence =
plant: age > 20
SUGGEST
( : status = suggested)

: occurrence = &
plantation : appearance = &
: status = cancelled
SUGGEST
( : status = suggested)

: occurrence = &
not(plantation : status = ) &
( : status = cancelled OR
: occurrence = )
SUGGEST
( : status = cancelled)

plantation : type = &
plant: age > 10 &
plant: status = &

```

```

not(      : occurrence =      )&
not(      : occurrence =      )
      SUGGEST
(      : status = suggested)

plantation : type =      &
plant: age > 10 &
plant: status =      &
(      : occurrence =      OR
      : occurrence =      )
      SUGGEST
(      : status = cancelled)

      : occurrence =      &

plantation : type =      &
plant: age > 10 &
plant: status =      &
not(      : occurrence =      ) &
not(      : occurrence =      )
      SUGGEST
(      : status = suggested)

      : occurrence =      &

plantation : type =      &
plant: age > 10 &
plant: status =      &
(      : occurrence =      OR
      : occurrence =      )
      SUGGEST
(      : status = cancelled)

      : occurrence =      &

plant: age > 20 days &
plant: status =      OR
(plant: age > 30 days &
plant: status =      )
SUGGEST

```

(: status = suggested)
 : occurrence = &
plant: age > 30 days &
 : no of occurrence = 1 &
 : application date <= Session date - 15 days

SUGGEST

(: status = suggested)
 : occurrence = &
plant: age > 30 days &
 : no of occurrence = 2 &
 : application date <= Session date - 15 days

SUGGEST

(: status = suggested)
 : occurrence = &
Plant: variety type = &
 : occurrence = &

plant: age > 20

SUGGEST

(: status = suggested)

Plantation: type = &
 : occurrence = &
 : occurrence = &

Session: date > 15/2

SUGGEST

(: status = suggested)

(plant: age > 50 &
plantation: crop type = &
plant: status = &

: no of occurrence < 11 &

SUGGEST

(: status = suggested)

```

        : occurrence =          &
plant: age > 70 &
plantation: crop type =          &
plant: status =          &
    SUGGEST
(
    : status = suggested)

        : occurrence =          &
plantation: crop type =          &
plant: age > 100 &
        : occurrence =          &
pods: status =
    SUGGEST
(
        : status = suggested)

```

domain-model: application model; % this model is new
part: tuple(apply);

axioms:

```

(Plantation: type =          OR
Plantation: type =          OR
Plantation: type =          )
    APPLY
Operation: name =          &
        : operation number = 1

(Plantation: type =          OR
Plantation: type =          OR
Plantation: type =          )
    APPLY
Operation: name =          &
        : operation number = 2

(Plantation: type =          OR
Plantation: type =          OR
Plantation: type =          ) &

```

(soil: type = OR
previous crop : fertilizer =)
 APPLY
Operation: name = &
 : operation number = 3

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
 APPLY
Operation: name = &
 : operation number = 4

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
 APPLY
Operation: name = &
 : operation number = 5

Irrigation: type = &
 APPLY
Operation: name = &
 : operation number = 6

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
 APPLY
Operation: name = &
 : operation number = 7

Irrigation: type =
 SUGGEST
Operation: name = &
 : operation number = 8

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
 APPLY
Operation: name = &
 : operation number = 9

Irrigation: type =
 APPLY
Operation: name = &
 : operation number = 10

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
 APPLY
Operation: name = &
 : operation number = 11

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
 APPLY
Operation: name = &
 : operation number = 12

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
 APPLY
Operation: name = &
 : operation number = 13

Plantation: type =
 APPLY
Operation: name = &
 : operation number = 14

Plantation: type =
APPLY
Operation: name = &
: operation number = 15

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
APPLY
Operation: name = &
: operation number = 16

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
APPLY
Operation: name = &
: operation number = 17

(Plantation: type = OR
Plantation: type = OR
Plantation: type =)
APPLY
Operation: name = &
: operation number = 18

Plant: variety type = &
APPLY
Operation: name = &
: operation number = 19

(plantation : type = OR
plantation : type =)
APPLY
Operation: name = &
: operation number = 20

(plantation : type = OR
 plantation : type =)
 APPLY
 Operation: name = &
 : operation number = 21

(Plantation: type = OR
 Plantation: type = OR
 Plantation: type =)
 APPLY
 Operation: name = &
 : operation number = 22

Plantation: type =
 APPLY
 Operation: name = &
 : operation number = 23

(Plantation: type = OR
 Plantation: type = OR
 Plantation: type =)
 APPLY
 Operation: name = &
 : operation number = 24

plantation: crop type = &
 APPLY
 Operation: name = &
 : operation number = 25

domain-model: assignment model;
part: tuple(assign);

The most of this model is updated, so the following is the complete assignment model

axioms:

: status = suggested &
ASSIGN

: method =

: status = suggested &
ASSIGN

: method = 25 - 20

: status = suggested &
(plantation: type = OR
plantation: type =)

ASSIGN

: method = 25 - 20

: status = suggested &
plantation: type =

ASSIGN

: method = 25 - 20

: status = suggested &
(plantation: type = OR

plantation: type =)

ASSIGN

: method = "

"

: status = suggested &
plantation: type =

ASSIGN

: method =

,"

```

: status = suggested &
(plantation: type = OR
plantation: type = )
ASSIGN
: method = "

: status = suggested &
plantation: type =
ASSIGN
: method =

"

: status = suggested &
ASSIGN
: method = "
" &

: status = suggested &
plantation: type =
ASSIGN
: method = " 120 - 100 "

: status = suggested &
plantation: type =
ASSIGN
: method = " 80 - 60 "

: status = suggested &
plantation: type =
ASSIGN
: method = " 10 "

: status = suggested
ASSIGN

```

```

: method = - 7 "
" 10

: status = suggested &
Irrigation: type =
ASSIGN
: method = " "

: status = suggested &
Irrigation: type =
ASSIGN
: method = - - "
" -

: status = suggested &
Irrigation: type =
ASSIGN
: method = " ( ) "

: status = suggested &
Irrigation: type =
ASSIGN
: method = " 2-1 "

: status = suggested
ASSIGN
: method = " "
" / 4 25

: status = suggested &
plantation: type =
ASSIGN
: method =
" 30-20

```

: status = suggested &
 (plantation: type = OR
 plantation: type =)
 ASSIGN
 : method = "

: status = suggested &
 plantation: type = &
 soil: type =
 ASSIGN
 : method = 15 "

3 - 2

" 3 "

: status = suggested &
 plantation: type = &
 soil: type =
 ASSIGN
 : method = 10 "

3 - 2

" 3 "

: status = suggested &
 plantation: type =
 soil: type =
 ASSIGN

: method = 7-5

3 - 2

" 3 "

: status = suggested &
plantation: type = &
soil: type =

ASSIGN

: method = 15 "

3 - 2

" 4 "

: status = suggested &
plantation: type = &
soil: type =

ASSIGN

: method = 10 "

3 - 2

" 4 "

: status = suggested &
plantation: type =
soil: type =

ASSIGN
: method = 7-5

3 - 2

"
" 4
: status = suggested &
plantation: type = &
soil: type =
ASSIGN
: method = 15 "

3 - 2

" 5 "
: status = suggested &
plantation: type = &
soil: type =
ASSIGN
: method = 10 "

3 - 2

" 5 "
: status = suggested &
plantation: type = &
soil: type =

ASSIGN

: method = 7-5

3 - 2

"

" 5

: status = suggested

ASSIGN

: method = 2 -1.5

"

"

50

: status = suggested &

plantation: type =

ASSIGN

: method = "

"

: status = suggested &

plantation: type =

ASSIGN

: method =

2

"

""

2

: status = suggested &

Planting: type =

ASSIGN

: method =

"

"

: status = suggested &

Planting: type =

&

ASSIGN

: method =

"

"


```

: status = suggested
  ASSIGN
: method = "
"

: status = suggested
  ASSIGN
: method = "
"

: status = suggested &
  ASSIGN
: method = "
"

: status = suggested
  ASSIGN
: method = "
"

: status = suggested
  ASSIGN
: method = 2 -1.5 "
" 50

: status = suggested
  ASSIGN
: method = "
"

: status = suggested
  ASSIGN
: method = 300 1,25 "
"

: status = suggested &
plantation: crop type =
  ASSIGN

```

```

: method = "
( )
: status = suggested &
plantation: crop type =
ASSIGN
: method = "
: status = suggested
ASSIGN
: method = "

```

domain-model: prediction model;
parts: tuple(predict)

The following is the new model.

axioms:

```

plant: age > 10 &
plant: age <= 20
    PREDICT
(plant: possible status = [ ] &
plantation: possible status = [ ] )

```

```

plant: age > 20 &
plant: age <= 30
    PREDICT
(plant: possible status = [ ] &
plantation: possible status = [ ] )

```

```

plant: age > 30 &
plant: age <= 40
    PREDICT
(plant: possible status = [ ] )

```

```

plant: age > 40 &
plant: age <= 50
    PREDICT
(plant: possible status = [ ] )

```

```

plant: age > 50 &

```

plant: age <= 90 &
: occurrence =
PREDICT
(plant: possible status = [])

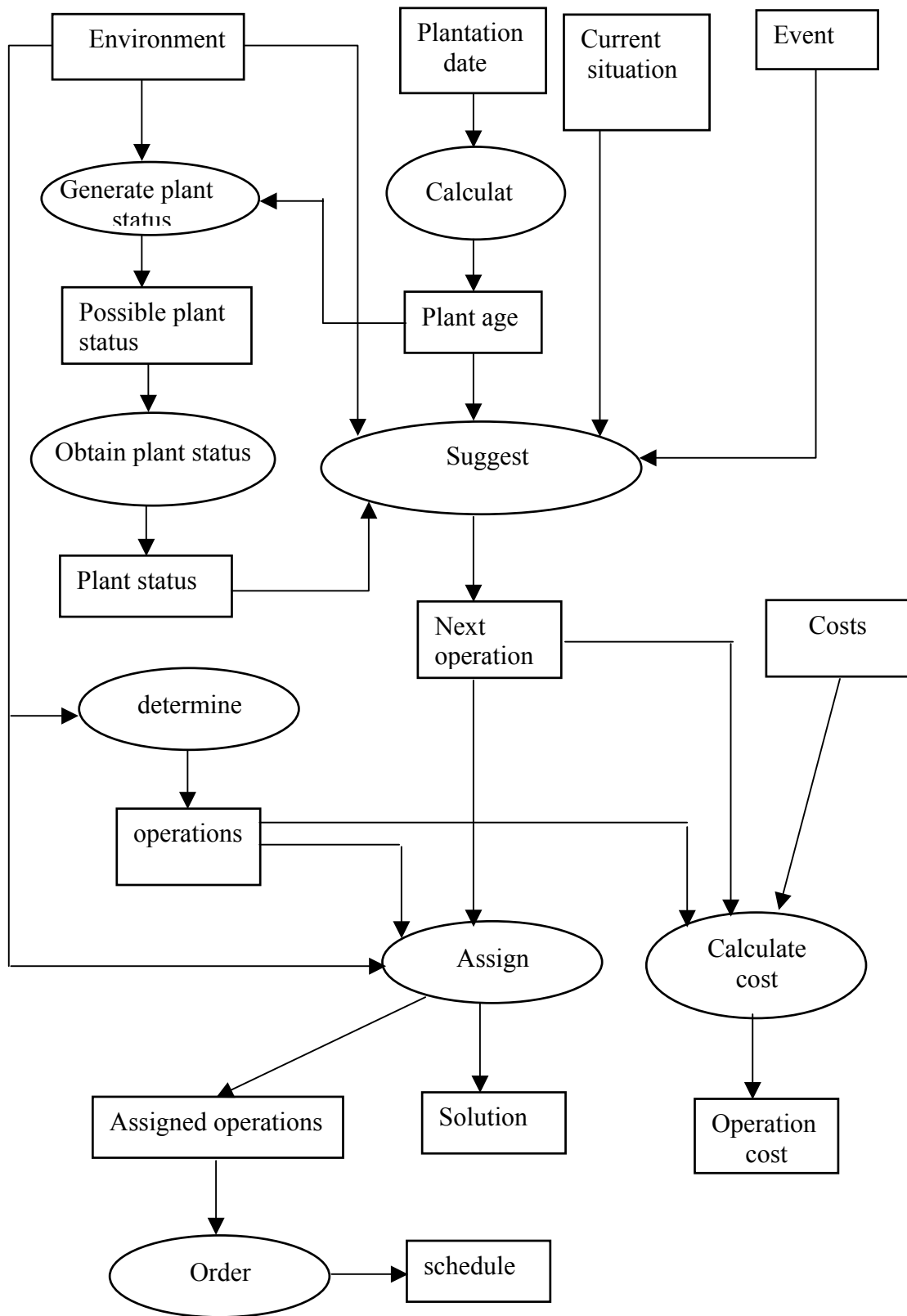
plant: age > 70 &
plant: age <= 120 &
plantation: crop type = &
: occurrence =
PREDICT
(plant: possible status = [])

plant: age > 100 &
plantation: crop type = &
: occurrence = &
PREDICT
(plant: possible status = [])

3. Inference Knowledge

3.1 Inference Structure

The inference structure is typical as in the technical report 'Strawberry Plant Care Design' number TR/CLAES/245/2002.9, it is shown in the following figure.



3.2 Inference specification

inference: calculate

operation-type: calculate the plant age.

input-roles: plantation date.

output-roles: plant age.

static-roles: there is no static roles

spec: plant age is calculated in days by subtracting the plantation date from the session date.

inference: generate plant status

operation-type: generates the possible plant status.

input-roles: environment, plant age.

output-roles: possible plant status.

static-roles: PREDICT \in prediction–model.

spec: the possible plant status are generated by applying "PREDICT " relation.

inference: obtain plant status

operation-type: transfer task.

input-roles: possible plant status.

output-roles: plant status.

static-roles: there is no static roles.

spec: obtain the current plant status from the user.

inference: suggest

operation-type: suggest the next agricultural operation.

input-roles: environment, plant age, event, plant status, accomplished operation, plantation date.

output-roles: next operation.

static-roles: SUGGEST \in suggestion–model.

spec: the next agricultural operation are to be suggested by applying " SUGGEST " relation.

inference: determine

operation-type: determine the agricultural operations schedule.

input-roles: environment.

output-roles: operations.

static-roles: APPLY \in application model.

spec: the agricultural operations schedule is determined by applying " APPLY" relation.

inference: assign

operation-type: assign parameters to the suggested operation.

input-roles: next operation, plantation date.

output-roles: solution.

static-roles: ASSIGN \in assignment–model.

spec: assign the method to the suggested operation by applying " ASSIGN " relation.

inference: order

operation-type: order the operations ascending according to operation: operation number .

input-roles: assigned operations

output-roles:. schedule

static-roles: there is no static role

spec: get operation: name(L), /* L is a list */
for each element(E) in L get E: operation number,
sort ascending L according to operation number of each element,
the sorted list is L',
DISPLAY(L')

inference: calculate cost

operation-type: calculate cost of the suggested operation.

input-roles: next operation, costs.

output-roles: operation cost.

static-roles: there is no static role

spec:

```
IF the user request is (next operation)
THEN
  Begin
    IF the method of the next operation is mechanical
    THEN get the tool name (T) of the next operation
          get the tool hiring rate (THR) of (T)
          get the total time/Feddan (TT) of applying the next operation
          get the area (A) of the plantation
          cost = THR*TT*A

    IF the method of the next operation is manual
    THEN get the number of men/Feddan (M) of applying the next
operation
          get the labor wage (W)
          get the area (A) of the plantation
          cost = M*W*A

  End
ELSE
  Begin
    get operation: name (N)          % N is a list
    While N <> {}
    Begin
      N = [E | Tail]
      IF the method of 'E' is mechanical
      THEN get the tool name (T) of 'E'
            get the hiring rate (HR) of (T)
            get the total time/Feddan (TT) of applying 'E'
            get the area (A) of the plantation
            Cost = HR*TT*A
            Insert Cost in E: cost
```

```

        TotalCost = TotalCost + Cost
        N = Tail
    IF the method of 'E' is manual
    THEN
        IF E: labor age = men
        THEN
            get number of men per feddan( (No) of 'E'
            get man wage ( ) (W)
        IF E: labor age = boys
        THEN
            get number of boys per feddan( (No) of 'E'
            get boy wage ( ) (W)

        get the area (A) of the plantation
        Cost = No*W*A
        Insert Cost in E: cost
        TotalCost = TotalCost + Cost
        N = Tail
    End

```

4. Task Knowledge

The task knowledge is typical as in the technical report 'Strawberry Plant Care Design' number TR/CLAES/245/2002.9

task: bean plant care;

task-definition:

goal: suggest the next agricultural operation;

input: Environment: {soil: type, plant: status, plantation: type, plant density, appearance, crop type};

plantation date: {plantation: date}

event: {event: value}

output: solution: { suggested operation, importance, method}

task-body:

type: composite

subtasks: calculate, generate plant status, obtain plant status, suggest, assign

additional-roles:

Possible plant status: {plant: possible status, pods: possible status}

Plant status {plant: status}

plant age: {plant: age}

Next operation {operation: status = suggested}

control-structure:

```

OBTAIN (plantation: date)                                % from data base
calculate (PD: plantation date) → PA: plant age),
OBTAIN (last suggested operation (OP))                  % from database,
IF plant: age > 10 & plant: age <= 50
THEN BEGIN

```



```

        generate plant status(E: environment → P: possible plant status),
        obtain plant status(P → PS: plant status), % using screen in fig. 1
        OBTAIN (event: occurrence) % from user by using screen in fig. 1
    END
IF plant: age > 50
THEN BEGIN
    generate plant status(E: environment → P: possible plant status),
    obtain plant status(P → PS: plant status), % using screen in fig. 2
    OBTAIN (event: occurrence) % from user by using screen in fig. 2
    END
OBTAIN (OP: occurrence) % from user
Assert (OP: occurrence) % in database
IF (OP: occurrence = OR ) & OP: importance =
THEN BEGIN
    prompt the user
        " " "OP " "
    END
ELSE BEGIN
    suggest(E, PA, EV, PS → NO: next operation),
    assert(NO) in database,
    assign(NO, E → Sol:solution),
    PRESENT(Sol)
    END

```

task: bean plant care;

task-definition:

goal: suggest the next agricultural operation,

presents a schedule of the agricultural operations;

input: Environment: {soil: type, previous crop : fertilizer, Irrigation: type,
Planting: type };

plant status: { plant: status};

plant density: {plantation: plant density};

plantation date: {plantation: date};

current situation: {operation: occurrence};

event: {event: value};

cost: {the cost of equipments, labor , ... etc from database}

output: solution: { suggested operation, importance, method},

operation cost: {cost of the suggested operation},

schedule: {the schedule of the agricultural operation during the season},

task-body:

type: composite

subtasks: calculate, generate plant status, obtain plant status, suggest, assign,
determine, order

additional-roles:

Possible plant status: {plant: possible status }

Plant status {plant: status}

plant age: {plant: age}

Next operation {operation: status = suggested}

Operations {operation: name}

control-structure:

IF the user request: value =

```
THEN
BEGIN
  OBTAIN (last suggested operation (OP))           % from database,
  OBTAIN (OP: occurrence)                         % from user
  Assert (OP: occurrence)                         % in database
  IF (OP: occurrence =          OR          ) & OP: importance =
  THEN BEGIN
    prompt the user
    "          "OP "          "
  END
ELSE BEGIN
  IF (OP: occurrence =          ) & operation cost: value =
  THEN prompt the user to type the actual cost of OP
    Assert(OP: actual cost)                       % in database
  IF (          : occurrence =          )
  THEN BEGIN
    IF plantation: date is UNKNOWN
    THEN BEGIN
      OBTAIN (plantation: date), % from the user
      END
      calculate(PD: plantation: date → PA: plant: age)
      IF plant: age > 10
      THEN BEGIN
        generate plant status(E: environment →
          P: possible plant status),
        obtain plant status(P → PS: plant status)
        END
        IF ((plantation : type =          &
          plant: age > 10 )OR          : occurrence =          )
        THEN BEGIN OBTAIN (event: occurrence) END
        END
        suggest(E, CS: current situation, PA, EV: event, PS →
          NO: next operation),
        IF NO is empty
        THEN
          Prompt the user ‘          ’
        ELSE
        BEGIN
          assert(NO) in database,
          assign(NO, E → Sol: solution),
          IF operation cost: value =
          THEN calculate cost(NO, C: costs → OC: operation cost)
```

```

                PRESENT(Sol, OC)
            ELSE
                PRESENT(Sol)

        END
    END
END
ELSE IF the user request: value =
    THEN determine(E: environment → O: operations),
        assign(E, O → AP: assigned operations),
        order(AP → S: schedule),
        IF operation cost: value =
            THEN calculate cost(O, C: costs → OC: operation cost)
                PRESENT(S, OC)
            ELSE PRESENT(S)

```

5. Database

The data base of this version is exactly similar to database of strawberry plant care expert system. after updating the legal values.

- Restrictions on values:

plantation: type = → irrigation: type = &
 planting: type =

planting: type = → irrigation: type =

irrigation: type = → planting: type =

soil: type = → irrigation: type =

6. User Interface

The user interface is similar to the user interface of the strawberry expert system.

7. Test Cases

Case 1

Input

Session date: 15/10/2002

user request:

Plantation: type =

soil type:

irrigation type:

planting type:

previous crop : fertilizer =

seeds: status =

Plant: variety type = &

plantation: crop type =

Output

:

:

:

Case 2

Input

Session date: 18/10/2002

user request:

: occurrence =

Output

:

:

25 - 20

:

Case 3

Input

Session date: 19/10/2002

user request:

: occurrence =

Output

" :
" :
" :

Case 4

Input

Session date: 19/10/2002

user request:

: occurrence =

Output

:
:
:

Case 5

Input

Session date: 20/10/2002

user request:

: occurrence =

Output

:

:

25 - 20

:

Case 6

Input

Session date: 21/10/2002

user request:

: occurrence =

Output

:

:

120 - 100

"

:

Case 7

Input

Session date: 21/10/2002

user request:

: occurrence =

Output

:
:
:

Case 8

Input

Session date: 21/10/2002

user request:

: occurrence =

Output

:
:
:

25

" / 4

Case 9

Input

Session date: 22/10/2002

user request:

: occurrence =

Output

20

:

:

:

30-

Case 10

Input

Session date = 23/10/2002

: occurrence =

Output

15

:

:

:

" 5

"

Case 11

Input

: occurrence =
Plantation date = 23/10/2002
Session date = 5/11/2002
plant: status =
Plantation: status =

Output

"

:

:

:

:

:

:

"

:

:

:

Case 12

Input

: occurrence =
Session date = 6/11/2002
plantation : status = &
: occurrence =

Output

:
:
:
"

Case 13

Input

: occurrence =
Session date = 14/11/2002
plantation : status = &

Output

_____ :
:
"
:

Case 14

Input

: occurrence =
Session date = 15/11/2000
Plant: variety type = &

Output

:
:
:

Case 15

Input

Session date = 16/11/2003
plant: status =

Output

300 1,25 :

Case 16

Input

Session date = 24/11/2003
: no of occurrence = 1
: application date = 16/11/2003

Output

300 1,25 :

Case 17

Input

Session date = 2/12/2003
: no of occurrence = 2
: application date = 24/11/2003

Output

:

300

1,25

:

:

Case 18

Input

Session date = 25/12/2003

plant: status =

Output

:

:

:

"()

APPENDIX I

Agricultural Operations Knowledge

:

:

:

:

:

1 :

1 :

:

:

:

:

4-3 :

:

:

:

:

:

:

:

:

:

:

1 :

2 :

:

:

:

:

5-1 :

:

/ / 4

/ / 0.5

:

:

25 - 20 :

:

+

:
 :
 :
 :
 :
 1 :
 3 :
 :
 :
 :
 :
 0.5 :
 / . 4
 / .
 :
 / 12 / 20
 2.5
 / 1.5 /
 :
 :
 :

:

:

:

:

:

1 :

4 :

:

:

:

:

:

/ . 2

/ . 0.5

:

:

:

:

:

:

:

:

:

1 :

5 :

:

:

:

:

/ / 4 :

:

:

25 - 20

:

:

:

:

:

:

:

1 :

6 :

:

:

:

:

:

6/ . 1

10 / . 1

:

:

" :

"

:

:

:

:

:

:

1 :

7 :

:

:

:

:

:

4/ . 1

10 / . 1

:

:

:

10

120-100

80-60

:

:

:

:

:

:

:

1 :

8 :

:

:

:

:

:

:

:

:

/ . 4 / / / 2 :

:

:

:

:

:

:

10 - 7

+

:

:

:

:

:

:

1 :

7 :

:

:

:

:

/ . 6

8 / . 6

:

:

:

-

-

:

:

:

:

:

:

:

1 :

10 :

:

:

:

:

:

:

:

/ . 4 :

18 16

:

/ 4

25

:

:

:

:

:

:

" / 4

25

:

:

:

:

:

:

:

1 :

12 :

:

:

:

:

/ 4 :

:

:

:

:

-

2-1

:

-

()

:

:

:

:

:

:

1 :

9 :

:

:

:

6 :

10

::

/ . 4

(540)

2 / . 1

::

:

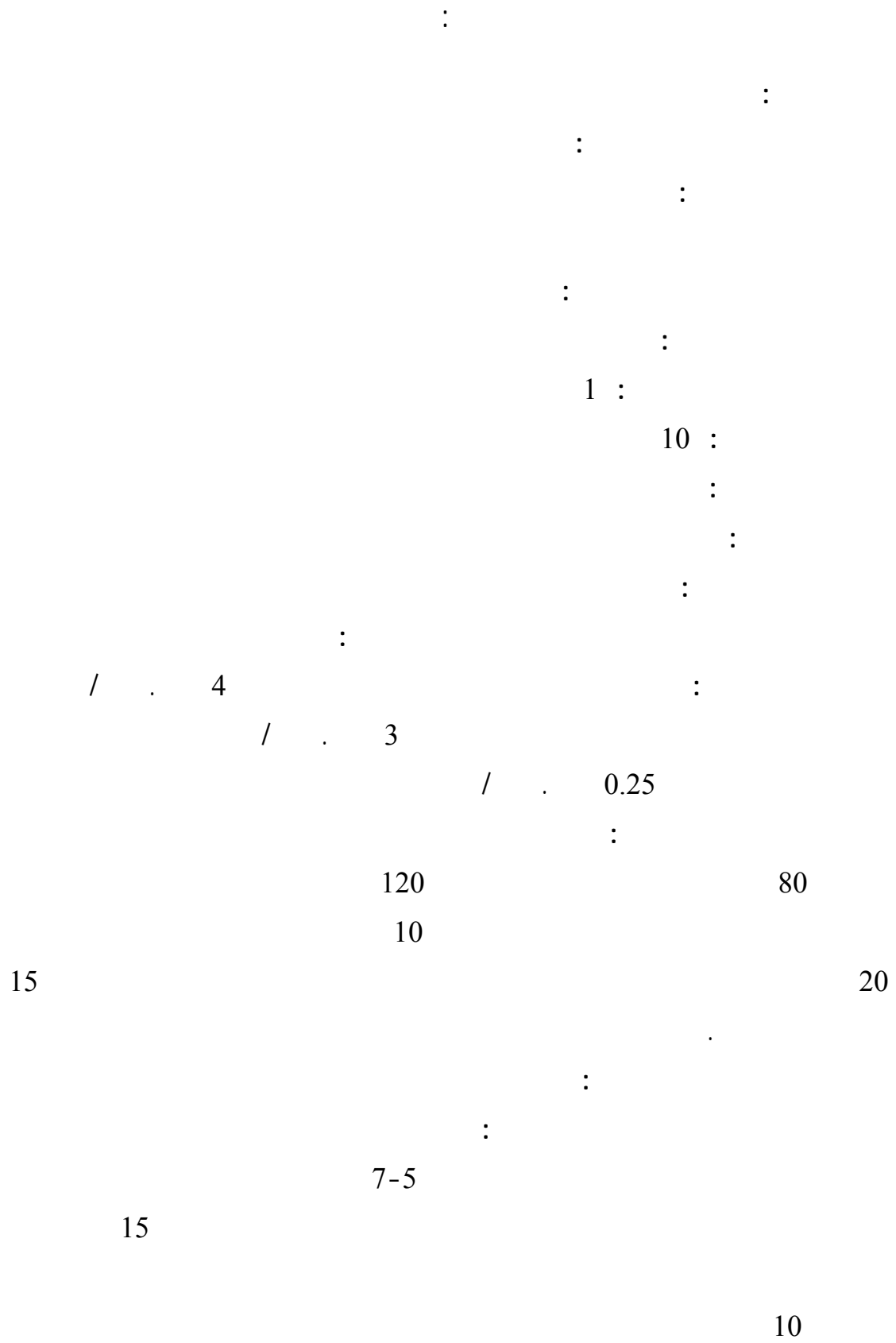
:

30-20

:

.

:



3 - 2

5

4

3

:

.

:

:

:

:

:

3 -1 :

16 :

:

:

:

.

:

/ . 4 :

:

:

:

-

"

"

:

-

"

"

:

:

:

:

:

:

1 :

14 :

:

:

:

:

/ . 6 :

/ 300 50 :

:

" 50 2 -1.5 "

:

:

:

:

:

:

:

17 :

:

:

:

:

/ . 4 :

:

:

:

"

:

:

:

:

:

:

1 :

22 :

:

:

:

:

/ . 6 :

:

:

:

"

:

:

:

:

:

:

1 :

14 :

:

:

:

30 – 21 :

/ . 1 ::

()

:

()

:

:

2

2

:

3/ . 1
/ . 2
/ . 3

:

:

:

:

:

1 :

14 :

:

:

:

::

:

:

:

:

:

/ . 2
/ . 1
/ . 1

+ () :

12

:

:

:

:

:

:

3-1 :

17 :

:

:

:

:

:

:

/ . 1 :

:

8/ . 1 :

:

:

:

:

:

:

:

:

:

:

:

:

:

3-2 :

18 :

:

:

& & :

:

:

/ . 2

/ . 3

2/ . 1

:

/ 300/ 1,25

8/ 300/ 1,25

:

1

300 1,25 :

:

8

:

:

:

:

:

:

10

20 :

:

:

:

:

:

/ . 8-6

/ . 10-8

/ . 3

/ . 10

:

:

:

()

:

:
 :
 :
 :
 :
 :
 1 :
 21 :
 :
 :
 :
 :
 :
 / . 3 :
 :
 :
 :
 :