Integrated Implementation
Of
Diagnosis, Treatment and Database
For
Citrus Expert Systems
(CITEX4)

Contents

1. Introduction

The objective of this report is to represent the implementation of integrated Citrus Expert system (CITEX4) including two sub expert systems: diagnosis and treatment in addition to other one other sub system database. This implementation is based on the integrated design report (TR/CLAES/194/2001.1). This system is implemented using KROL99 tools that support building the concepts, rules, inference, and task code.

The following eight sections represent the implementation code of the common knowledge base, diagnosis expert system, treatment expert system, database system, and user interface system.

2. Comments on implementation

- The legal value silver is added property f-color for concept fruit.
- The legal value coarse is added property f-shape for concept fruit.
- The legal value drop is added property f-r_status for concept fruit.
- The legal value >=1 <=52 is replaced by >=1 <=52 to property current_month for concept plant.
- The legal value pring is replaced by spring to property season for concept plant.
- The concept insects in page 3 in the design is replaced by insect in the implementation.
- The relation 'Disorder VERIFY Disorder' is not implemented according to the …
- All CAUSED_BY relations in the design is implemented as one relation.
- All CONFIRM relations in the design implemented in one relation.
- All VERIFY relations in the design implemented in one relation.
- The property current_date for concept Plant is added.
- The concept environmental in design is replaced by environment in the implementation.
- The table 'select_table' is added in the implementation with the following fileds:-

  Sid numeric
  gid numeric
  did numeric
  fid numeric

- Condation for check the method is adder to all the rules in "Treat_Op ENHANCED_BY Treat_Op"
- The following rules in Treat_Op ENHANCED_BY Treat_Op relation

aphids #citrus_white_fly			aphids	Advice	The pressure of spraying motor must not exceed 100 pound per square inch without direct application Spray the infested trees only.

citrus_white_fly #aphids			citrus_white_fly	Advice	The pressure of spraying motor must not exceed 100 pound per square inch without direct application Spray the infested trees only.
Aphids & citrus_white_fly			Aphids	Advice	Spray infested trees only. The pressure of spraying motor must not exceed 100 pound per square inch without direct application. This operation used as shared treatment for aphids and citrus white fly.
Aphids & citrus_white_fly			citrus_white_fly	Advice	Spray infested trees only. The pressure of spraying motor must not exceed 100 pound per square inch without direct application. This operation used as shared treatment for aphids and citrus white fly.

Are modify to

aphids citrus_white_fly	Method method	chemical spray =\ chemical spray	aphids	Advice	The pressure of spraying motor must not exceed 100 pound per square inch without direct application Spray the infested trees only.
citrus_white_fly aphids	Method Method	chemical spray =\ chemical spray	citrus_white_fly	Advice	The pressure of spraying motor must not exceed 100 pound per square inch without direct application Spray the infested trees only.
Aphids & citrus_white_fly	Method Method	chemical spray chemical spray	Aphids	Advice	Spray infested trees only. The pressure of spraying motor must not exceed 100 pound per square inch without direct application. This operation used as shared treatment for aphids and citrus white fly.
Aphids & citrus_white_fly	Method method	chemical spray chemical spray	citrus_white_fly	Advice	Spray infested trees only. The pressure of spraying motor must not exceed 100 pound per square inch without direct application. This operation used as shared treatment for aphids and citrus white fly.

3. Common Domain layer

3.1 Concept Prosperities

The following code is the ontology implementation.

```
/* File name : c_concept.pl */
:-ensure_loaded('$KROL/lib/inferenc').
farm_data :: {
        concept_description('') &
        attributes([ sid([]), gid([]), did([]), fid([]), month([]) ]) &
        type(sid/1, integer) &
        ul(sid/1, 10) &
        ll(sid/1, 1) &
        prompt(sid/1, 'Enter the sector ID', []) &
```

```
        necessary(sid/1) &
        type(gid/1, integer) &
        ul(gid/1, 1000) &
        ll(gid/1, 1) &
        prompt(gid/1, 'Enter the governorate ID', []) &
        necessary(gid/1) &
        type(did/1, integer) &
        ul(did/1, 1000) &
        ll(did/1, 1) &
        prompt(did/1, 'Enter the directorate ID', []) &
        necessary(did/1) &
        type(fid/1, integer) &
        ul(fid/1, 1000) &
        ll(fid/1, 1) &
        prompt(fid/1, 'Enter the farm ID', []) &
        necessary(fid/1) &
        type(month/1, integer) &
        ul(month/1, 12) &
        ll(month/1, 1) &
        prompt(month/1, 'Enter the month', []) &
        necessary(month/1) &
        super(domain_class)
}.

variety :: {
        concept_description('') &
        attributes([ value([]) ]) &
        type(value/1, nominal) &
        source_of_value(value/1,
[database(citex4ds,farm_table(_157883,_157885,_157887,_157889,_157891,_157893,_157895,_1578
97,_157899,_157901,_157903,_157905,_157907,_157909,_157911,Vvariety_name,_157915),Vvariety
_name)]) &
        legal(value/1, [ lime, navel, succar, valencia ]) &
        necessary(value/1) &
        super(domain_class)
}.
plantation :: {
        concept_description('') &
        attributes([existence([]), current_date([]),plantation_date([])]) &
        type(existence/1, nominal) &
        legal(existence/1, [ yes, no]) &
        type(current_date/1, date) &
        type(plantation_date/1, date) &
        source_of_value(plantation_date/1,
[database(citex4ds,farm_table(_28920,_28922,_28924,_28926,_28928,_28930,Vplantation_date,_2893
4,_28936,_28938,_28940,_28942,_28944,_28946,_28948,_28950,_28952),Vplantation_date)]) &
        super(domain_class)
}.
soil :: {
        concept_description('') &
        attributes([ca_carbonate([]),ec([]),ph([]), water_table_level([]) ]) &
        type(ca_carbonate/1, real) &
        source_of_value(ca_carbonate/1,[database(citex4ds,soil_assessment_table(_39972,_39974,_3
9976,_39978,_39980,_39982,_39984,_39986,Vca_carbonate,_39990,_39992,_39994),Vca_carbonate)]
) &
        ul(ca_carbonate/1, 14.0) &
        ll(ca_carbonate/1, 0.1) &
        type(ec/1, real) &
        source_of_value(ec/1,database(citex4ds,soil_table(_45050,_45052,_45054,_45056,_45058,Ve
c,_45062,_45064,_45066),Vec)]) &
```

```
        ul(ec/1, 14.0) &
        ll(ec/1, 0.1) &
        type(ph/1, real) &
        source_of_value(ph/1,database(citex4ds,soil_table(_56836,_56838,_56840,_56842,_56844,_5
6846,Vph,_56850,_56852),Vph)]) &
        ul(ph/1, 14.0) &
        ll(ph/1, 0.1) &

        type(water_table_level/1, real) &
        source_of_value(water_table_level/1,database(citex4ds,soil_table(_67908,_67910,_67912,_67
914,Vwater_table_level,_67918,_67920,_67922,_67924),Vwater_table_level)]) &
        ul(water_table_level/1, 10.4) &
        ll(water_table_level/1, 0.1) &
        super(domain_class)
}.
water :: {
        concept_description('') &
        attributes([        eciw([])  ]) &
        type(eciw/1, real) &
        source_of_value(eciw/1,database(citex4ds,water_table(_82100,_82102,_82104,Veciw),Veciw
)]) &
        ul(eciw/1, 5) &
        ll(eciw/1, 0.01) &
        super(domain_class)
}.
leaves :: {  concept_description('') &
        attributes([l_color([]),l_shape([]),l_status([]),l_type([]) l_c_position([]) ]) &
        type(l_color/1, nominal) &
        multiple(l_color/1) &
        prompt(l_color/1, 'What is the leaves shape ?', []) &
        legal(l_color/1, [green,    green_network,  light_green,dark_green,    geen_to_red,
                yellow,  brown,  black,    purple,  bronze   ]) &
        type(l_shape/1, nominal) &
        multiple(l_shape/1) &
        prompt(l_shape/1, 'What is the leaves shape?', []) &
        legal(l_shape/1, [normal,curled,    webbed,honey_dew,cup_shape, unsimilar_blade_halves,
        zigzag_tunnels    ]) &
        type(l_status/1, nominal) &
        multiple(l_status/1) &
        prompt(l_status/1, 'What is the leaves status?', []) &
        legal(l_status/1, [normal,drop,insect_persent,small,wilted]) &
        type(l_type/1, nominal) &
        multiple(l_type/1) &
        prompt(l_type/1, 'What is the age of the infected leaves?', []) &
        legal(l_type/1, [   new_leaves,old_leaves     ]) &
        type(l_c_position/1, nominal) &
        multiple(l_c_position/1) &
        prompt(l_c_position/1, 'Where is the position of the infestation on the leaves?', []) &
        legal(l_c_position/1, [entire_leaf,inverted_v,'lower surface','upper surface',
        'outer edge','leaf base','leaf margin',veins,'between veins',        'main veins','leaf tip']) &
        super(domain_class)
}.
leaf_spots :: {
        concept_description('') &
        attributes([existence([]),    l_s_color([]),l_s_shap([]),l_s_position([])]) &
        type(existence/1, nominal) &
        prompt(existence/1, 'Are they any sopts on leaves ?', []) &
        legal(existence/1, [yes,    no]) &
        type(l_s_color/1, nominal) &
        multiple(l_s_color/1) &
```

```
        prompt(l_s_color/1, 'What is color of the spots on leaves?', []) &
        legal(l_s_color/1, [yellow,brown,dusty,  silver,  rust,black]) &
        type(l_s_shap/1, nominal) &
        multiple(l_s_shap/1) &
        prompt(l_s_shap/1, 'What is the shape of the spots on leaves ?', []) &
        legal(l_s_shap/1, [raised,sunken,necrotic,'zigzag tunnels','concentric zones']) &
        type(l_s_position/1, nominal) &
        multiple(l_s_position/1) &
        prompt(l_s_position/1, 'What is the position of the spots on leaves?', []) &
        legal(l_s_position/1, [scattered,  'upper surface',  'lower surface',  'between veins',
                'between veins of lower surface',  'midrib upper surface']) &
        super(domain_class)
}.
fruits :: {
        concept_description('') &
        attributes([f_c_position([]),f_color([]),f_shape([]),f_r_status([])  ]) &
        type(f_c_position/1, nominal) &
        multiple(f_c_position/1) &
        prompt(f_c_position/1, 'What is the position of the infestation on the fruit?', []) &
        legal(f_c_position/1, [  'entire fruit',  'styler end'  ]) &
        necessary(f_c_position/1) &
        type(f_color/1, nominal) &
        multiple(f_color/1) &
        prompt(f_color/1, 'What is the fruits color?', []) &
        legal(f_color/1, [black,  green,  'green styler end',normal,purple,  rust,
        yellow,  'yellow styler end',silver  ]) &
        necessary(f_color/1) &
        type(f_shape/1, nominal) &
        multiple(f_shape/1) &
        prompt(f_shape/1, 'What is the fruit shape?', []) &
        legal(f_shape/1, [asymatric,cracks,malformed,normal,small,soft,  coarse  ]) &
        necessary(f_shape/1) &
        type(f_r_status/1, nominal) &
        multiple(f_r_status/1) &
        prompt(f_r_status/1, 'What is the fruits status?', []) &
        legal(f_r_status/1, [creasing,irregular,leathery,normal,reduced,rough,  'rough and
thickened',thickened,  thin,drop]) &
        necessary(f_r_status/1) &
        super(domain_class)
}.
fruit_spots :: {
        concept_description('') &
        attributes([existence([]),  f_s_color([]),f_s_position([]),f_s_shape([])]) &
        type(existence/1, nominal) &
        multiple(existence/1) &
        prompt(existence/1, 'Are there spots on fruit? ', []) &
        legal(existence/1, [yes,  no]) &
        necessary(existence/1) &
        type(f_s_color/1, nominal) &
        multiple(f_s_color/1) &
        prompt(f_s_color/1, 'What is the color of the spots on the fruit?', []) &
        legal(f_s_color/1, [green,yellow,brown,  red,sliver,bronze,'scabby patches']) &
        necessary(f_s_color/1) &
        type(f_s_position/1, nominal) &
        multiple(f_s_position/1) &
        prompt(f_s_position/1, 'What is the position of the spots on the fruit?', []) &
        legal(f_s_position/1, [scattered,'any position',rind,'stiller and stem ends',
                'fruits facing the sun']) &
        necessary(f_s_position/1) &
        type(f_s_shape/1, nominal) &
```

```
        multiple(f_s_shape/1) &
        prompt(f_s_shape/1, 'What is the shape of the spots on the fruit?', []) &
        legal(f_s_shape/1, [circular,irregular,raised,coarse,'large and circular',
                'gum pocket','zigzag tunnels']) &
        necessary(f_s_shape/1) &
        super(domain_class)
}.
flowers :: {
        concept_description('') &
        attributes([fl_color([]),fl_status([]),f_l_shape([])        ]) &
        type(fl_color/1, nominal) &
        multiple(fl_color/1) &
        prompt(fl_color/1, 'What is the flowers color?', []) &
        legal(fl_color/1, [normal,brown,     yellow    ]) &
        necessary(fl_color/1) &
        type(fl_status/1, nominal) &
        multiple(fl_status/1) &
        prompt(fl_status/1, 'What is the flowers color?', []) &
        legal(fl_status/1, [nromal,drop        ]) &
        necessary(fl_status/1) &
        type(f_l_shape/1, nominal) &
        multiple(f_l_shape/1) &
        prompt(f_l_shape/1, 'What is the flowers shape?', []) &
        legal(f_l_shape/1, [normal,aggregated,         eaten      ]) &
        necessary(f_l_shape/1) &
        super(domain_class)
}.
branches :: {
        concept_description('') &
        attributes([b_type([]),b_status([]),b_color([])]) &
        type(b_type/1, nominal) &
        multiple(b_type/1) &
        prompt(b_type/1, 'What is the age of the infected branches ?', []) &
        legal(b_type/1, [flushes,    'old growth']) &
        necessary(b_type/1) &
        type(status/1, nominal) &
        multiple(status/1) &
        prompt(b_status/1, 'What is the branches status ?', []) &
        legal(b_status/1, [decline,'die back',dry,        flattened,'insect present',                normal,
        stunted,  thickened,'gray fellvet'       ]) &
        necessary(b_status/1) &
        type(b_color/1, nominal) &
        multiple(b_color/1) &
        prompt(b_color/1, 'What is the branches color?', []) &
        legal(b_color/1, [black,     brown,   normal,  pale,rust,'spotted yellowish']) &
        necessary(b_color/1) &
        super(domain_class)
}.
trunk :: {
        concept_description('') &
        attributes([t_shape([]),      t_position([])       ]) &
        type(t_shape/1, nominal) &
        multiple(t_shape/1) &
        prompt(t_shape/1, 'What is the trunk shape ?', []) &
        legal(t_shape/1, [normal,'fungal growths','lichen growths','bark scaling',             'gum
spots',dwarfing    ]) &
        necessary(t_shape/1) &
        type(t_position/1, nominal) &
        multiple(t_position/1) &
        prompt(t_position/1, 'What is the trunk position ?', []) &
```

```
        legal(t_position/1, ['basal part',      'feeder roots'      ]) &
        necessary(t_position/1) &
        super(domain_class)
}.
buds :: {
        concept_description('') &
        attributes([u_color([]),u_shape([]),u_status([])          ]) &
        type(u_color/1, nominal) &
        multiple(u_color/1) &
        prompt(u_color/1, 'What is the buds color ?', []) &
        legal(u_color/1, [normal,   brown    ]) &
        necessary(u_color/1) &
        type(u_shape/1, nominal) &
        multiple(u_shape/1) &
        prompt(u_shape/1, 'What is the buds shape ?', []) &
        legal(u_shape/1, [rosette,deformed]) &
        necessary(u_shape/1) &
        type(u_status/1, nominal) &
        multiple(u_status/1) &
        prompt(u_status/1, 'What is the buds status?', []) &
        legal(u_status/1, [normal,abnormal]) &
        necessary(u_status/1) &
        super(domain_class)
}.

roots :: {
        concept_description('') &
        attributes([r_color([]),      r_status([]),r_type([])]) &
        type(r_color/1, nominal) &
        multiple(r_color/1) &
        prompt(r_color/1, 'What is the root color ?', []) &
        legal(r_color/1, [normal,brown,      black     ]) &
        necessary(r_color/1) &
        type(r_status/1, nominal) &
        multiple(r_status/1) &
        prompt(r_status/1, 'What is the root status ?', []) &
        legal(r_status/1, [normal,'fungal growth',      sloughing,necrotic,adhesive]) &
        necessary(r_status/1) &
        type(r_type/1, nominal) &
        multiple(r_type/1) &
        prompt(r_type/1, 'What is the type of the infected roots ?', []) &
        legal(r_type/1, [   'main roots','feeder roots']) &
        necessary(r_type/1) &
        super(domain_class)
}.
twigs :: {
        concept_description('') &
        attributes([tw_color([]),     tw_shape([]),tw_status([])]) &
        type(tw_color/1, nominal) &
        multiple(tw_color/1) &
        prompt(tw_color/1, 'What is the twigs color ?', []) &
        legal(tw_color/1, [brown, rust         ]) &
        necessary(tw_color/1) &
        type(tw_shape/1, nominal) &
        multiple(tw_shape/1) &
        prompt(tw_shape/1, 'What is the twigs shape?', []) &
        legal(tw_shape/1, [eaten    ]) &
        necessary(tw_shape/1) &
        type(tw_status/1, nominal) &
        multiple(tw_status/1) &
```

```
        prompt(tw_status/1, 'What is the twigs status ?', []) &
        legal(tw_status/1, [        dieback  ]) &
        necessary(tw_status/1) &
        super(domain_class)
}.
plant :: {
        concept_description('') &
        attributes([current_date([]),age([]),season([]),current_week([]),current_month([])        ]) &
        type(current_date/1, date) &
        type(age/1, real) &
        ul(age/1, 50) &
        ll(age/1, 0) &
        prompt(age/1, '', []) &
        type(yield/1, real) &
        type(season/1, nominal) &
        source_of_value(season/1, [table(plant_determine_plant)]) &
        legal(season/1, [spring,    summer,autumn,winter    ]) &
        type(current_week/1, integer) &
        source_of_value(current_week/1, [derived(treated_by)]) &
        ul(current_week/1, 53) &
        ll(current_week/1, 1) &
        type(current_month/1, integer) &
        ul(current_month/1, 12) &
        ll(current_month/1, 1) &
        prompt(current_month/1, 'What is the current month?', []) &
        necessary(current_month/1) &
        super(domain_class)
}.
operation :: {
        concept_description('') &
attributes([material_qty([]),unit([]),material_name([]),method([]),material_gr1([]),
material_gr2([]),material_gr3([]),material_gr4([]),material_gr5([]),material_gr6([]),
        material_gr7([]),material_gr8([]),material_gr9([]),material_gr10([]),
        material_gr11([]),material_gr12([])]) &
        type(material_qty/1, real) &
        source_of_value(material_qty/1, [derived(treat_op_determine_treat_op)]) &
        ul(material_qty/1, 1000) &
        ll(material_qty/1, 0) &
        target(material_qty/1, '') &
        type(unit/1, nominal) &
        source_of_value(unit/1, [derived(treat_op_determine_treat_op)]) &
        legal(unit/1, [     'L/100 l water',    'gm/1 l water',     'gm/100 l water','gm/tree',
        'kg CuSo4 + 2 Kg CaO + 10 L water','ml + 25 ml/100 l water', 'ml/100 l water',
'Kg Cu So4 + 1.5 CaO/100 l water','ml + 150 ml/100 l water','ml + 250 ml/100 L water',
        'kg/feddan','L/feddan','ml + L/100 l water','ml + 500 ml/100 l water',
        'kg/100 l water',   'as below']) &
        type(material_name/1, nominal) &
        multiple(material_name/1) &
        source_of_value(material_name/1, [[derived(treated_by)]]) &
        legal(material_name/1, ['K.Z. 95%','Kimisol 95%','actellic 50%',        'agro oil 80%',
        aikaten,  'ammonium nitrate','anthio 33%','bolum oil 80%','bordeaux past',
        'calcium chloride','calcium nitrate','caprimex 98%','copox 50%',copper_oxychloride,
        'cuprus K.Z 50%','focal oil 82%','furidan 10%','halomac 65%','libacid 50% +bominal',
        magnesium_sulfate,'malathion 57%','malthion 57% + policure','misrona oil 80%',
        'micro element mixture',    'neron 50%',none,'ortis 5% sc + kz oil','pory coper 50%',
        potassium_nitrate,potassium_permenganat,potassium_sulfate,pride,'pro coper 50%',
        'ragbi 10%','royal oil 80%','super aside','super masrona 94%','super royal 95%',
        'temic 15%',topsin,'triple phosphate',urea,vaydete,'vertimec + K.Z oil 95%',
'vertimec + Kimisol oil 95%','vertimec + super masrona 94%','vertimec + super royal oil 95%','vertimec
1.8%','vertimec 1.8% + kz oil'        ]) &
```

```
        target(material_name/1, '') &
        type(method/1, nominal) &
        source_of_value(method/1, [derived(treated_by)]) &
        legal(method/1, [advice,    'chemical spray',disinfection,'foliage nutrition',painting,
        'soil treatment'     ]) &
        target(method/1, '') &
        type(material_gr1/1, nominal) &
        prompt(material_gr1/1, 'Select available material', []) &
        legal(material_gr1/1, ['K.Z. 95%','Kimisol 95%',        'super masrona 94%',
'super royal 95%']) &
        necessary(material_gr1/1) &
        type(material_gr2/1, nominal) &
        prompt(material_gr2/1, 'Select available material', []) &
        legal(material_gr2/1, ['actellic 50%',aikaten,'anthio 33%',        'super aside']) &
        necessary(material_gr2/1) &
        type(material_gr3/1, nominal) &
        prompt(material_gr3/1, 'Select available material', []) &
legal(material_gr3/1,['caprimex 98%','copox 50%',copper_oxychloride,'cuprus K.Z 50%','halomac
65','pory coper 50%','pro coper 50%']) &
        necessary(material_gr3/1) &
        type(material_gr4/1, nominal) &
        prompt(material_gr4/1, 'Select available material', []) &
        legal(material_gr4/1, ['agro oil 80%','bolum oil 80%','focal oil 82%',                'misrona oil
80%','royal oil 80%'         ]) &
        necessary(material_gr4/1) &
        type(material_gr5/1, nominal) &
        prompt(material_gr5/1, 'Select available material', []) &
        legal(material_gr5/1, ['vertimec + K.Z oil 95%',        'vertimec + Kimisol oil 95%',
        'vertimec + super masrona 94%',     'vertimec + super royal oil 95%'      ]) &
        necessary(material_gr5/1) &
        type(material_gr6/1, nominal) &
        prompt(material_gr6/1, 'Select available material', []) &
        legal(material_gr6/1, ['neron 50%', 'ortis 5% sc + kz oil','vertimec 1.8% + kz oil']) &
        necessary(material_gr6/1) &
        type(material_gr7/1, nominal) &
        prompt(material_gr7/1, 'Select available material', []) &
        legal(material_gr7/1, ['ortis 5% sc + kz oil',pride,'vertimec 1.8% + kz oil'          ]) &
        necessary(material_gr7/1) &
        type(material_gr8/1, nominal) &
        prompt(material_gr8/1, 'Select available material', []) &
        legal(material_gr8/1, ['furidan 10%','ragbi 10%','temic 15%']) &
        necessary(material_gr8/1) &
        type(material_gr9/1, nominal) &
        prompt(material_gr9/1, 'Select available material', []) &
        legal(material_gr9/1, [      urea,      'ammonium nitrate']) &
        necessary(material_gr9/1) &
        type(material_gr10/1, nominal) &
        prompt(material_gr10/1, 'Select available material', []) &
        legal(material_gr10/1, [     potassium_nitrate,potassium_sulfate]) &
        necessary(material_gr10/1) &
        type(material_gr11/1, nominal) &
        prompt(material_gr11/1, 'Select available material', []) &
        legal(material_gr11/1, [     'calcium chloride','calcium nitrate']) &
        necessary(material_gr11/1) &
        type(material_gr12/1, nominal) &
        prompt(material_gr12/1, 'Select available material', []) &
        legal(material_gr12/1, [     'ibacid 50% + bominal','malthion 57% + policure']) &
        necessary(material_gr12/1) &
        super(domain_class)
}.
```

```
treat_op :: {
        concept_description('') &
        attributes([tool([]),application_time([]),advice([]),date([]),number([]),  special_date([])  ]) &
        type(tool/1, nominal) &
        multiple(tool/1) &
        prompt(tool/1, '', []) &
        legal(tool/1, [manual,'sprayer motor']) &
        type(application_time/1, nominal) &
        multiple(application_time/1) &
        source_of_value(application_time/1, [[derived(treat_op_determine_treat_op)]]) &
        legal(application_time/1, ['any suitable time','early morning or afternoon'          ]) &
        target(application_time/1, '') &
        type(advice/1, nominal) &
        multiple(advice/1) &
        source_of_value(advice/1, [[derived(enhanced_by)]]) &
        legal(advice/1, [  'Also, avoid excess irrigation water near the trunk','Application of acaricides
is recommended at 20 % infestation, in general. Spot spraying localized infestation is good practice and
tractor drawn equipment with agitator is often the ideal machine for application. Spraying should be as
a mist, tacking umbrella shape at lower pressure and as possible over the entire tree','Avoid excess of
nitrogen ertilizers and organic manure near the trunk. Also, avoid excess irrigation water near the
trunk.','Collect infected fruits and bury it. Perform the suitable agriculture practices',
        'Collect infested fruits and bury it.','Control the insects that produce the honey dew',
'Cultivate plant tarps for scarab like faba-beans, turnip and cauliflower','Good caring the diseased trees;
i.e. better agriculture practices and fertilization to extend the productive life of tree when yield becomes
not economic, the diseased trees must be replaced. Use certified transplants','Improve the agriculture
practices','Improve the growth of trees to protect the fruits from direct sun light', 'Increase quantity of
fertilizer application by 25% and incrementally increased up to 50% or until disappearance of nutrient
deficiency observations, then apply the recommendations given by the fertilization expert system',
        'Increase quantity of fertilizer application by 50% and incrementally increased up to 100% or
until disappearance of nutrient deficiency observations, then apply the recommendations given by the
fertilization expert system','Infected young trees should be replaced by other healthy plants. Use
certified transplants','It is important to check the soil salinity, and in case of high salinity the leaching is
recommended','Lichens control includes good agricultural practices; i.e. pruning and avoid excess
irrigation water',  'Manage the irrigation and increase the fertilization quantity of Potassium','No foliage
application during the flowering stage and fruit setting','No foliage application during the fruits
collecting period','No significant response of trees to foliar application during winter. Therefore treat
your trees in the beginning of spring','No treatment for this pest, such that it is not important
economically','No treatment for this phenomena where its economic importance is limited','Picking up
the insects twice a day','Remove fungal growths and painting the wound by Bordeaux past then spray
the green area of trees.  The formula of Bordeaux past is:  1 kg cuso + 2 kg cad  + water','Spot spraying
localized infestation is good practice and tractor drawn equipment with agitator is often the ideal
machine for application','Spray the infested trees only','Spray trees of entire farm','Spraying should be
as a mist, tacking umbrella shape at lower pressure and as far as possible',          'Spraying should be as
a mist, tacking umbrella shape at lower pressure and as far as possible from downwards to up words
and pointing to the core of tree',      'Spraying should be as a mist, tacking umbrella shape at lower
pressure and as far as possible from upwards to downwards',   'Spraying two branches only in each tree
and collects infested fruits and bury it.','Spread watercolor traps at the rate 35 to 40 traps per
feddan','Substitute the nitrogen quantity in the fertilization expert system recommendation by its
equivalence of calcium nitrate','The diseased trees must be replaced','The gum pocked must be removed
with sharp knife, the wound and exposed tissues must be disinfected with solution','The micro elements
mixture is formulated, for every 100 lt water , as follow :  150 gm Iron Chelate (EDTA) + 30 gm Zinc
Chelate + 15 Mang. Chelate  + 6 gm Copper Sulfate + 30 Magnesium Sulfate + 0.3 gm Borax','The
micro elements mixture is formulated, for every 100 lt water, as follow :  30 gm Iron Chelate (EDTA)
+ 30 gm Zinc Chelate + 75 Mang. Chelate  + 6 gm Copper Sulfate + 30 Magnesium Sulfate + 0.3 gm
Borax','The micro elements mixture is formulated, for every 100 lt water, as follow:  30 gm Iron
Chelate (EDTA) + 150 gm Zinc Chelate + 15 Mang. Chelate  + 6 gm Copper Sulfate + 30 Magnesium
Sulfate + 0.3 gm Borax','The pressure of spraying motor must not exceed 100 pound per square inch
without direct application','The treatment at this time is not recommended.','The treatment at this time
is not recommended. Time of chemical control in late of April, in case of infestation','The treatment at
this time is not recommended. Time of chemical control in late of February, in case of infestation','The
```

treatment at this time is not recommended. Time of chemical control in late of May, in case of infestation','The trees must be completely washed','This operation used as shared treatment for aphids and citrus white fly','Use compost organic manure','Use fit spraying motor with good mixing. The trees must be completely washed.','Use irrigation program to add leaching requirements','You must follow this operation by light irrigation to avoid application of fruit bearing trees'

```
        ]) &
        target(advice/1, '') &
        type(date/1, date) &
        source_of_value(date/1, [derived(treated_by)]) &
        target(date/1, '') &
        type(number/1, integer) &
        source_of_value(number/1, [derived(treated_by)]) &
        ul(number/1, 50) &
        ll(number/1, 1) &
        target(number/1, '') &
        type(special_date/1, nominal) &
        source_of_value(special_date/1, [derived(treated_by)]) &
        legal(special_date/1, ['next 1/12','next 1/2','next 1/6','next 22/6','next 13/7', 'next
22/2','next summer','next winter']) &
        super(operation)
}.
disorder :: {
        concept_description('') &
        attributes([suspected([]),
        confirmed([]),
        highly_confirmed([]),
        iron_def_sp([]),
        manganese_def_sp([]),
        zinc_def_sp([]),
        nitrogen_def_sp([]),
        salt_injury_sp([]),
        magnesium_def_sp([]),
        calcium_def_sp([]),
        potassium_def_sp([]),
        phosphorus_infestation([]),
        nitrogen_infestation([]),
        potassium_infestation([])]) &
        type(suspected/1, nominal) &
        multiple(suspected/1) &
        source_of_value(suspected/1, [[derived(caused_by_disoders)]]) &
        legal(suspected/1, [psorosis,impieetratura,stubborn,anthracnose, gummosis,
        sooty_mold,wilt_root_rot,black_root_rot,ganoderma_rot,brown_rot,
        alternaria_rot,armillaria_root_rot,alternaria_leaves_spot, gum_spots,
        sun_burn,fruit_cracking,fruit_creasing, lichens, rose_scarab,
        mediterranean_fruit_fly, citrus_white_fly,scales, aphids, citrus_flower_moth,
        mealy_bug,green_stink_bug,leafminer, rust_mite,bud_mite,
        brown_mite,flat_mite,citrus_nematude, nitrogen_def,phosphorus_def,
        potassium_def, magnesium_def,manganese_def,iron_def,calcium_def,
        zinc_def,salt_injury]) &
        type(confirmed/1, nominal) &
        multiple(confirmed/1) &
        source_of_value(confirmed/1, [[derived(confirm_disorders)]]) &
        legal(confirmed/1, [psorosis,impieetratura,stubborn,anthracnose, gummosis,
        sooty_mold,wilt_root_rot,black_root_rot,ganoderma_rot,brown_rot,
        alternaria_rot,armillaria_root_rot,alternaria_leaves_spot, gum_spots,
        sun_burn,fruit_cracking,fruit_creasing, lichens, rose_scarab,
        mediterranean_fruit_fly, citrus_white_fly,scales, aphids, citrus_flower_moth,
        mealy_bug,green_stink_bug,leafminer, rust_mite,bud_mite,brown_mite,
        flat_mite,citrus_nematude,nitrogen_def, phosphorus_def,potassium_def,
        magnesium_def,manganese_def,iron_def,calcium_def,zinc_def,salt_injury]) &
```

```
        type(highly_confirmed/1, nominal) &
        multiple(highly_confirmed/1) &
        source_of_value(highly_confirmed/1, [[derived(verify_disorders)]]) &
        legal(highly_confirmed/1, [psorosis,impieetratura,stubborn,anthracnose,  gummosis,
        sooty_mold,wilt_root_rot,black_root_rot,ganoderma_rot,brown_rot,
        alternaria_rot,armillaria_root_rot,alternaria_leaves_spot,  gum_spots,sun_burn,
        fruit_cracking,fruit_creasing,lichens,rose_scarab,mediterranean_fruit_fly,
        citrus_white_fly,scales,  aphids,  citrus_flower_moth,mealy_bug,  green_stink_bug,
leafminer,rust_mite,bud_mite,brown_mite,flat_mite,citrus_nematude,nitrogen_def,phosphorus_def,pot
assium_def,magnesium_def,manganese_def,iron_def,calcium_def,
        zinc_def,salt_injury]) &
        type(iron_def_sp/1, nominal) &
        source_of_value(iron_def_sp/1, [user]) &
prompt(iron_def_sp/1, 'What is the spread range of the iron defection infestation ?', []) &
        legal(iron_def_sp/1, ['most trees']) &
        necessary(iron_def_sp/1) &
        type(manganese_def_sp/1, nominal) &
        source_of_value(manganese_def_sp/1, [user]) &
prompt(manganese_def_sp/1, 'What is the spread range of the manganese defection infestation ?', []) &
        legal(manganese_def_sp/1, ['most trees'  ]) &
        necessary(manganese_def_sp/1) &
        type(zinc_def_sp/1, nominal) &
        source_of_value(zinc_def_sp/1, [user]) &
prompt(zinc_def_sp/1, 'What is the spread range of the zinc defection infestation ?', []) &
        legal(zinc_def_sp/1, ['most trees']) &
        necessary(zinc_def_sp/1) &
        type(nitrogen_def_sp/1, nominal) &
        source_of_value(nitrogen_def_sp/1, [user]) &
prompt(nitrogen_def_sp/1, 'What is the spread range of the nitrogen defection infestation ?', []) &
        legal(nitrogen_def_sp/1, ['most trees']) &
        necessary(nitrogen_def_sp/1) &
        type(salt_injury_sp/1, nominal) &
        source_of_value(salt_injury_sp/1, [user]) &
prompt(salt_injury_sp/1, 'What is the spread range of the salt_injury defection infestation ?', []) &
        legal(salt_injury_sp/1, [   'most trees']) &
        necessary(salt_injury_sp/1) &
        type(magnesium_def_sp/1, nominal) &
        source_of_value(magnesium_def_sp/1, [user]) &
prompt(magnesium_def_sp/1, 'What is the spread range of the magnesium defection infestation ?', [])
&
        legal(magnesium_def_sp/1, [  'most trees']) &
        necessary(magnesium_def_sp/1) &
        type(calcium_def_sp/1, nominal) &
        source_of_value(calcium_def_sp/1, [user]) &
prompt(calcium_def_sp/1, 'What is the spread range of the calcium defection infestation ?', []) &
        legal(calcium_def_sp/1, ['most trees']) &
        necessary(calcium_def_sp/1) &
        type(potassium_def_sp/1, nominal) &
        source_of_value(potassium_def_sp/1, [user]) &
prompt(potassium_def_sp/1, 'What is the spread range of the potassium defection infestation ?', []) &
        legal(potassium_def_sp/1, ['most trees']) &
        necessary(potassium_def_sp/1) &
        type(phosphorus_infestation/1, nominal) &
        source_of_value(phosphorus_infestation/1, [user]) &
      prompt(phosphorus_infestation/1, 'What is the Value of Phosphours Infestation? ', []) &
        legal(phosphorus_infestation/1, [low,'very low']) &
        necessary(phosphorus_infestation/1) &
        type(nitrogen_infestation/1, nominal) &
        source_of_value(nitrogen_infestation/1, [user]) &
        prompt(nitrogen_infestation/1, 'What is the Value of Nitrogen Infestation? ', []) &
```

```
        legal(nitrogen_infestation/1, [        low,      'very low']) &
        type(potassium_infestation/1, nominal) &
        source_of_value(potassium_infestation/1, [user]) &
        prompt(potassium_infestation/1, 'What is the Value of Potassium Infestation? ', []) &
        legal(potassium_infestation/1, [      low,      'very low']) &
        necessary(potassium_infestation/1) &
        super(treat_op)
}.
insects :: {
        concept_description('') &
        attributes([i_color([]),i_status([])]) &
        type(i_color/1, nominal) &
        multiple(i_color/1) &
        prompt(i_color/1, 'What is the insects color ?', []) &
        legal(i_color/1, [green,     black,white,red,   purple]) &
        necessary(i_color/1) &
        type(i_status/1, nominal) &
        multiple(i_status/1) &
        prompt(i_status/1, 'What is the insects status ?', []) &
        legal(i_status/1, [stationary,flying,stucked,aggregated]) &
        necessary(i_status/1) &
        super(domain_class)
}.
insect :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
disease :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
lichens :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.

mites :: {
        concept_description('') &
        attributes([          ]) &
        super(disorder)
}.
nematode :: {
        concept_description('') &
        attributes([          ]) &
        super(disorder)
}.
nutrition_def :: {
        concept_description('') &
        attributes([]) &
        super(disorder)
}.
virus :: {
        concept_description('') &
        attributes([]) &
        super(disease)
}.
fungal :: {
```

```
        concept_description('') &
        attributes([]) &
        super(disease)
}.
environmental :: {
        concept_description('') &
        attributes([]) &
        super(disease)
}.
psorosis :: {
        concept_description('') &
        attributes([        ]) &
        super(virus)
}.
impieetratura :: {
        concept_description('') &
        attributes([        ]) &
        super(virus)
}.
stubborn :: {
        concept_description('') &
        attributes([]) &
        super(virus)
}.
anthracnose :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
gummosis :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
sooty_mold :: {
        concept_description('') &
        attributes([        ]) &
        super(fungal)
}.
ganoderma_rot :: {
        concept_description('') &
        attributes([        ]) &
        super(fungal)
}.
alternaria_rot :: {
        concept_description('') &
        attributes([        ]) &
        super(fungal)
}.
armillaria_root_rot :: {
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
wilt_root_rot :: {
        concept_description('') &
        attributes([        ]) &
        super(fungal)
}.
alternaria_leaves_spot :: {
```

```
        concept_description('') &
        attributes([]) &
        super(fungal)
}.
gum_spots :: {
        concept_description('') &
        attributes([        ]) &
        super(fungal)
}.
sun_burn :: {
        concept_description('') &
        attributes([        ]) &
        super(environmental)
}.
fruit_cracking :: {
        concept_description('') &
        attributes([        ]) &
        super(environmental)
}.
fruit_creasing :: {
        concept_description('') &
        attributes([        ]) &
        super(environmental)
}.
salt_injury :: {
        concept_description('') &
        attributes([        ]) &
        super(environmental)
}.
rose_scarab :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
mediterranean_fruit_fly :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
citrus_white_fly :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
scales :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
aphids :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
citrus_flower_moth :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
mealy_bug :: {
```

```
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
green_stink_bug :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
leafminer :: {
        concept_description('') &
        attributes([        ]) &
        super(insect)
}.
rust_mite :: {
        concept_description('') &
        attributes([        ]) &
        super(mites)
}.
bud_mite :: {
        concept_description('') &
        attributes([        ]) &
        super(mites)
}.
brown_mite :: {
        concept_description('') &
        attributes([        ]) &
        super(mites)
}.
flat_mite :: {
        concept_description('') &
        attributes([        ]) &
        super(mites)
}.
citrus_nematude :: {
        concept_description('') &
        attributes([        ]) &
        super(nematode)
}.
nitrogen_def :: {
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
phosphorus_def :: {
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
potassium_def :: {
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
magnesium_def :: {
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
manganese_def :: {
```

```
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
iron_def :: {
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
calcium_def :: {
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
zinc_def :: {
        concept_description('') &
        attributes([        ]) &
        super(nutrition_def)
}.
navel :: {
        concept_description('') &
        attributes([        ]) &
        super(variety)
}.
succar :: {
        concept_description('') &
        attributes([        ]) &
        super(variety)
}.
valencia :: {
        concept_description('') &
        attributes([        ]) &
        super(variety)
}.
lime :: {
        concept_description('') &
        attributes([        ]) &
        super(variety)
}.
ganoderma_rot_op1 :: {
        concept_description('') &
        attributes([        ]) &
        super(ganoderma_rot)
}.
ganoderma_rot_op2 :: {
        concept_description('') &
        attributes([        ]) &
        super(ganoderma_rot)
}.
wilt_root_rot_op1 :: {
        concept_description('') &
        attributes([        ]) &
        super(wilt_root_rot)
}.
wilt_root_rot_op2 :: {
        concept_description('') &
        attributes([        ]) &
        super(wilt_root_rot)
}.
leafminer_op1 :: {
```

```
        concept_description('') &
        attributes([        ]) &
        super(leafminer)
}.
leafminer_op2 :: {
        concept_description('') &
        attributes([        ]) &
        super(leafminer)
}.
leafminer_op3 :: {
        concept_description('') &
        attributes([        ]) &
        super(leafminer)
}.
rust_mite_op1 :: {
        concept_description('') &
        attributes([        ]) &
        super(rust_mite)
}.
rust_mite_op2 :: {
        concept_description('') &
        attributes([        ]) &
        super(rust_mite)
}.
bud_mite_op1 :: {
        concept_description('') &
        attributes([        ]) &
        super(bud_mite)
}.
bud_mite_op2 :: {
        concept_description('') &
        attributes([        ]) &
        super(bud_mite)
}.
brown_mite_op1 :: {
        concept_description('') &
        attributes([        ]) &
        super(brown_mite)
}.
brown_mite_op2 :: {
        concept_description('') &
        attributes([        ]) &
        super(brown_mite)
}.
flat_mite_op1 :: {
        concept_description('') &
        attributes([        ]) &
        super(flat_mite)
}.
flat_mite_op2 :: {
        concept_description('') &
        attributes([        ]) &
        super(flat_mite)
}.
citrus_nematude_op1 :: {
        concept_description('') &
        attributes([        ]) &
        super(citrus_nematude)
}.
citrus_nematude_op2 :: {
```

```
        concept_description('') &
        attributes([        ]) &
        super(citrus_nematude)
}.
```

4. Diganosis subsystem

4.1 Relations Between Concept

The following code is the relation between concept implementation for diagnosis subsystem.

```
/* File name : diag_table.pl */
:-ensure_loaded('$KROL/lib/tab').
plant_determine_plant :: {
        p([plant-current_month] , [plant-season]) &
         t(1,winter) &      t(2,winter) &      t(3,spring) &      t(4,spring) &      t(5,spring) &
         t(6,summer) & t(7,summer) & t(8,summer) & t(9,autumn) & t(10,autumn) &
         t(11,winter) & t(12,winter) &super(table)
}.

/* File name : diag_rules.pl */
:- use_module(library(lists), [memberchk/2]).
:- ensure_loaded('$KROL/lib/rule_exp').
caused_by_disoders :: {
r1([suspected(gummosis)in disorder,suspected(citrus_nematude)in disorder,suspected(nitrogen_def)in
disorder,suspected(potassium_def)in disorder,suspected(magnesium_def)in
disorder,suspected(manganese_def)in disorder,suspected(iron_def)in
disorder,suspected(calcium_def)in disorder,suspected(zinc_def)in disorder,suspected(salt_injury)in
disorder]) if
l_color(yellow) in leaves &
r2([suspected(citrus_white_fly)in disorder,suspected(aphids)in disorder,suspected(mealy_bug)in
disorder]) if
        l_color(black) in leaves &
r3([suspected(phosphorus_def)in disorder,suspected(salt_injury)in disorder]) if
        l_color(purple) in leaves &
r4([suspected(psorosis)in disorder,suspected(aphids)in disorder]) if
        l_color(green) in leaves &
r5([suspected(rust_mite)in disorder]) if        l_color(brown) in leaves &
r6([suspected(phosphorus_def)in disorder]) if l_color(dark_green) in leaves &
r7([suspected(potassium_def)in disorder]) if l_color(bronze) in leaves &
r8([suspected(iron_def)in disorder]) if l_color(green_network) in leaves &
r9([suspected(gummosis)in disorder]) if l_color(light_green) in leaves &
r10([suspected(leafminer)in disorder]) if l_shape(zigzag_tunnels) in leaves &
r11([suspected(scales)in disorder,suspected(leafminer)in disorder,suspected(rust_mite)in
disorder,suspected(brown_mite)in disorder,suspected(flat_mite)in
disorder,suspected(manganese_def)in disorder]) if
        existence(yes) in leaf_spots &
r12([suspected(calcium_def)in disorder]) if  l_shape(cup_shape) in leaves &
r13([suspected(aphids)in disorder]) if l_shape(curled) in leaves &
r14([suspected(citrus_white_fly)in disorder,suspected(aphids)in disorder,suspected(mealy_bug)in
disorder]) if l_shape(honey_dew) in leaves &
r15([suspected(bud_mite)in disorder]) if u_color(brown) in buds,
        (       value(lime) in variety
        ;       value(navel) in variety
        ), ! &
r16([suspected(wilt_root_rot)in disorder]) if
        l_color(yellow) in leaves,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
```

```
        ), ! &
r17([suspected(anthracnose)in disorder,suspected(alternaria_leaves_spot)in
disorder,suspected(gum_spots)in disorder]) if
        existence(yes) in leaf_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r18([suspected(armillaria_root_rot)in disorder]) if
        r_status('fungal growth') in roots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r19([suspected(zinc_def)in disorder]) if
        age(_11364) in plant,       :(_11364>=5),
        t_shape(dwarfing) in trunk &
r20([suspected(lichens)in disorder]) if
        age(_12012) in plant,       :(_12012>=5),
        t_shape('lichen growths') in trunk &
r21([suspected(lichens)in disorder]) if
        age(_12660) in plant,       :(_12660>=5),
        b_color('spotted yellowish') in branches &
r22([suspected(gummosis)in disorder]) if
        age(_13308) in plant,       :(_13308>=5),
        t_shape('gum spots') in trunk &
r23([suspected(stubborn)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        (       f_color(normal) in fruits
        ;       f_color('green styler end') in fruits
        ), ! &
r24([suspected(citrus_flower_moth)in disorder]) if
        season(spring) in plant,
        age(_15153) in plant,       :(_15153>=5),
        f_l_shape(aggregated) in flowers &
r25([suspected(citrus_flower_moth)in disorder]) if
        season(spring) in plant,
        age(_15977) in plant,       :(_15977>=5),
        l_color(geen_to_red) in leaves &
r26([suspected(rose_scarab)in disorder,suspected(citrus_flower_moth)in disorder]) if
        season(spring) in plant,
        age(_16852) in plant,       :(_16852>=5),
        f_l_shape(eaten) in flowers &
r27([suspected(potassium_def)in disorder,suspected(salt_injury)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_17967) in plant,       :(_17967>=5),
        f_shape(small) in fruits &
r28([suspected(phosphorus_def)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_19031) in plant,       :(_19031>=5),
        f_r_status('rough and thickened') in fruits &
r29([suspected(ganoderma_rot)in disorder]) if
        age(_19705) in plant,       :(_19705>=5),
        t_shape('fungal growths') in trunk,
```

```
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r30([suspected(psorosis)in disorder]) if
        age(_20931) in plant,       :(_20931>=5),
        t_shape('bark scaling') in trunk,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r31([suspected(sooty_mold)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        l_color(black) in leaves,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r32([suspected(sooty_mold)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), !,
        b_color(black) in branches &
r33([suspected(sooty_mold)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), !,
        f_color(black) in fruits &
r34([suspected(brown_mite)in disorder,suspected(green_stink_bug)in
disorder,suspected(impieetratura)in disorder,suspected(mediterranean_fruit_fly)in
disorder,suspected(sun_burn)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r35([suspected(fruit_creasing)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_28213) in plant,       :(_28213>=5),
        f_r_status(creasing) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r36([suspected(fruit_cracking)in disorder]) if
```

```
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_29855) in plant,       :(_29855>=5),
        f_shape(cracks) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r37([suspected(alternaria_rot)in disorder]) if
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        value(navel) in variety,
        f_color('yellow styler end') in fruits &
super(rules)
}.

confirm_disorders :: {
r1([confirmed(stubborn)in disorder]) if
        suspected(stubborn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_color('green styler end') in fruits &
r2([confirmed(citrus_flower_moth)in disorder]) if
        suspected(citrus_flower_moth) in disorder,
        season(spring) in plant,
        l_color(geen_to_red) in leaves &
r3([confirmed(citrus_flower_moth)in disorder]) if
        suspected(citrus_flower_moth) in disorder,
        season(spring) in plant,
        (       f_l_shape(aggregated) in flowers
        ;       f_l_shape(eaten) in flowers
        ), ! &
r4([confirmed(rose_scarab)in disorder]) if
        suspected(rose_scarab) in disorder,
        season(spring) in plant,
        f_l_shape(eaten) in flowers &
r5([confirmed(phosphorus_def)in disorder]) if
        suspected(phosphorus_def) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_r_status('rough and thickened') in fruits &
r6([confirmed(potassium_def)in disorder]) if
        suspected(potassium_def) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_shape(small) in fruits,
        (       f_r_status(creasing) in fruits
        ;       f_r_status(thin) in fruits
        ), ! &
r7([confirmed(salt_injury)in disorder]) if
        suspected(salt_injury) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        f_shape(small) in fruits &
```

```
r8([confirmed(gummosis)in disorder]) if
        suspected(gummosis) in disorder,
        age(_39646) in plant,       :(_39646>=5),
        t_shape('gum spots') in trunk,
        (       t_position('basal part') in trunk
        ;       t_position('feeder roots') in trunk
        ), ! &
r9([confirmed(gummosis)in disorder]) if
        suspected(gummosis) in disorder,
        age(_40886) in plant,       :(_40886>=5),
        (       l_color(light_green) in leaves
        ;       l_color(yellow) in leaves
        ), !,
        l_c_position('main veins') in leaves &
r10([confirmed(lichens)in disorder]) if
        suspected(lichens) in disorder,
        age(_42102) in plant,       :(_42102>=5),
        t_shape('lichen growths') in trunk &
r11([confirmed(lichens)in disorder]) if
        suspected(lichens) in disorder,
        age(_42946) in plant,
        :(_42946>=5),
        b_color('spotted yellowish') in branches,
        b_status('gray fellvet') in branches &
r12([confirmed(mediterranean_fruit_fly)in disorder]) if
        suspected(mediterranean_fruit_fly) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        (       f_s_color(red) in fruit_spots
        ;       f_s_color(yellow) in fruit_spots
        ), !,
        f_s_position('any position') in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r13([confirmed(green_stink_bug)in disorder]) if
        suspected(green_stink_bug) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        f_s_color(yellow) in fruit_spots,
        f_s_position(scattered) in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r14([confirmed(impieetratura)in disorder]) if
        suspected(impieetratura) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        existence(yes) in fruit_spots,
        f_s_color(green) in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
```

```
        ), ! &
r15([confirmed(salt_injury)in disorder]) if
        suspected(salt_injury) in disorder,
        l_color(purple) in leaves &
r16([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        existence(yes) in leaf_spots,
        (       l_s_color(brown) in leaf_spots
        ;       l_s_color(rust) in leaf_spots
        ), !,
        l_s_position(scattered) in leaf_spots &
r17([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        l_color(brown) in leaves,
        (       tw_color(brown) in twigs
        ;       tw_color(rust) in twigs
        ), ! &
r18([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        (       tw_color(brown) in twigs
        ;       tw_color(rust) in twigs
        ), !,
        existence(yes) in leaf_spots &
r19([confirmed(brown_mite)in disorder]) if
        suspected(brown_mite) in disorder,
        existence(yes) in fruit_spots,
        f_s_color(brown) in fruit_spots,
        f_s_position('stiller and stem ends') in fruit_spots &
r20([confirmed(brown_mite)in disorder]) if
        suspected(brown_mite) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(dusty) in leaf_spots,
        l_s_position('midrib upper surface') in leaf_spots &
r21([confirmed(flat_mite)in disorder]) if
        suspected(flat_mite) in disorder,
        (       l_s_color(brown) in leaf_spots
        ;       l_s_color(silver) in leaf_spots
        ), !,
        l_s_shap(sunken) in leaf_spots,
        l_s_position('between veins of lower surface') in leaf_spots &
r22([confirmed(citrus_nematude)in disorder]) if
        suspected(citrus_nematude) in disorder,
        l_color(yellow) in leaves,
        l_c_position(entire_leaf) in leaves,
        b_status('die back') in branches,
        b_type(flushes) in branches &
r23([confirmed(leafminer)in disorder]) if
        suspected(leafminer) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(silver) in leaf_spots &
r24([confirmed(leafminer)in disorder]) if
        suspected(leafminer) in disorder,
        l_shape(zigzag_tunnels) in leaves &

r25([confirmed(aphids)in disorder]) if
        suspected(aphids) in disorder,
        (       l_shape(curled) in leaves
        ;       l_shape(honey_dew) in leaves
        ), !,
        l_status(insect_persent) in leaves,
```

```
        l_type(new_leaves) in leaves &

r26([confirmed(aphids)in disorder]) if
        suspected(aphids) in disorder,
        l_status(insect_persent) in leaves,
        l_type(new_leaves) in leaves,
        (       l_color(black) in leaves
        ;       l_color(green) in leaves
        ), ! &

r27([confirmed(citrus_white_fly)in disorder]) if
        suspected(citrus_white_fly) in disorder,
        l_color(black) in leaves,
        l_status(insect_persent) in leaves &
r28([confirmed(citrus_white_fly)in disorder]) if
        suspected(citrus_white_fly) in disorder,
        l_shape(honey_dew) in leaves,
        l_status(insect_persent) in leaves &
r29([confirmed(citrus_white_fly)in disorder]) if
        suspected(citrus_white_fly) in disorder,
        l_color(black) in leaves,
        l_shape(honey_dew) in leaves,
        l_c_position('upper surface') in leaves &
r30([confirmed(scales)in disorder]) if
        suspected(scales) in disorder,
        existence(yes) in leaf_spots,
        (       l_s_color(black) in leaf_spots
        ;       l_s_color(yellow) in leaf_spots
        ), !,
        (       l_s_position('lower surface') in leaf_spots
        ;       l_s_position('upper surface') in leaf_spots
        ), !,
        l_status(insect_persent) in leaves &
r31([confirmed(mealy_bug)in disorder]) if
        suspected(mealy_bug) in disorder,
        l_color(black) in leaves,
        l_status(insect_persent) in leaves,
        l_type(old_leaves) in leaves,
        b_status('insect present') in branches &
r32([confirmed(mealy_bug)in disorder]) if
        suspected(mealy_bug) in disorder,
        l_status(insect_persent) in leaves,
        l_type(old_leaves) in leaves,
        b_status('insect present') in branches,
        l_shape(honey_dew) in leaves &
r33([confirmed(mealy_bug)in disorder]) if
        suspected(mealy_bug) in disorder,
        l_color(black) in leaves,
        l_shape(honey_dew) in leaves &
r34([confirmed(iron_def)in disorder]) if
        suspected(iron_def) in disorder,
        (       l_color(green_network) in leaves
        ;       l_color(yellow) in leaves
        ), !,
        (       l_c_position(entire_leaf) in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(new_leaves) in leaves &
r35([confirmed(manganese_def)in disorder]) if
        suspected(manganese_def) in disorder,
```

```
        l_color(yellow) in leaves,
        l_c_position('between veins') in leaves,
        l_type(new_leaves) in leaves &
r36([confirmed(manganese_def)in disorder]) if
        suspected(manganese_def) in disorder,
        existence(yes) in leaf_spots,
        l_s_position('between veins') in leaf_spots &
r37([confirmed(nitrogen_def)in disorder]) if
        suspected(nitrogen_def) in disorder,
        l_color(yellow) in leaves,
        (       l_c_position(entire_leaf) in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(old_leaves) in leaves &
r38([confirmed(potassium_def)in disorder]) if
        suspected(potassium_def) in disorder,
        (       l_color(bronze) in leaves
        ;       l_color(yellow) in leaves
        ), !,
        (       l_c_position('leaf margin') in leaves
        ;       l_c_position('leaf tip') in leaves
        ), !,
        l_type(old_leaves) in leaves &
r39([confirmed(zinc_def)in disorder]) if
        suspected(zinc_def) in disorder,
        l_color(yellow) in leaves,
        l_c_position('between veins') in leaves,
        l_type(new_leaves) in leaves,
        b_status(stunted) in branches,
        l_shape(unsimilar_blade_halves) in leaves &
r40([confirmed(alternaria_leaves_spot)in disorder,confirmed(zinc_def)in disorder]) if
        (       suspected(alternaria_leaves_spot) in disorder
        ;       suspected(zinc_def) in disorder
        ), !,
        t_shape(dwarfing) in trunk,
        b_status(stunted) in branches,
        l_shape(unsimilar_blade_halves) in leaves &
r41([confirmed(alternaria_leaves_spot)in disorder,confirmed(phosphorus_def)in disorder]) if
        (       suspected(alternaria_leaves_spot) in disorder
        ;       suspected(phosphorus_def) in disorder
        ), !,
        suspected(phosphorus_def) in disorder,
        (       l_color(dark_green) in leaves
        ;       l_color(purple) in leaves
        ), !,
        (       l_c_position('leaf tip') in leaves
        ;       l_c_position('lower surface') in leaves
        ;       l_c_position('outer edge') in leaves
        ), !,
        (       l_type(new_leaves) in leaves
        ;       l_type(old_leaves) in leaves
        ), ! &
r42([confirmed(calcium_def)in disorder]) if
        suspected(calcium_def) in disorder,
        l_color(yellow) in leaves,
        (       l_c_position('leaf margin') in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(new_leaves) in leaves &
r43([confirmed(calcium_def)in disorder]) if
```

```
        suspected(calcium_def) in disorder,
        (       l_c_position('leaf margin') in leaves
        ;       l_c_position(veins) in leaves
        ), !,
        l_type(new_leaves) in leaves,
        l_shape(cup_shape) in leaves &
r44([confirmed(magnesium_def)in disorder]) if
        suspected(magnesium_def) in disorder,
        l_color(yellow) in leaves,
        (       l_c_position('between veins') in leaves
        ;       l_c_position(inverted_v) in leaves
        ;       l_c_position('leaf base') in leaves
        ;       l_c_position('outer edge') in leaves
        ), !,
        l_type(old_leaves) in leaves &
r45([confirmed(bud_mite)in disorder]) if
        suspected(bud_mite) in disorder,
        (       value(lime) in variety
        ;       value(navel) in variety
        ), !,
        u_color(brown) in buds,
        u_status(abnormal) in buds &
r46([confirmed(bud_mite)in disorder]) if
        suspected(bud_mite) in disorder,
        (       value(lime) in variety
        ;       value(navel) in variety
        ), !,
        u_color(brown) in buds,
        (       u_shape(deformed) in buds
        ;       u_shape(rosette) in buds
        ), ! &
r47([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        value(lime) in variety,
        existence(yes) in fruit_spots,
        f_s_color(sliver) in fruit_spots,
        f_s_shape(coarse) in fruit_spots &
r48([confirmed(rust_mite)in disorder]) if
        suspected(rust_mite) in disorder,
        value(lime) in variety,
        l_color(brown) in leaves,
        f_shape(coarse) in fruits &
r49([confirmed(alternaria_rot)in disorder]) if
        suspected(alternaria_rot) in disorder,
        value(navel) in variety,
        f_color('yellow styler end') in fruits &
r50([confirmed(sun_burn)in disorder]) if
        suspected(sun_burn) in disorder,
        existence(yes) in fruit_spots,
        f_s_color(brown) in fruit_spots,
        f_s_position('fruits facing the sun') in fruit_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r51([confirmed(wilt_root_rot)in disorder]) if
        suspected(wilt_root_rot) in disorder,
        l_color(yellow) in leaves,
        l_status(wilted) in leaves,
        (       value(navel) in variety
```

```
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r52([confirmed(alternaria_leaves_spot)in disorder]) if
        suspected(alternaria_leaves_spot) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(yellow) in leaf_spots,
        (       l_s_position('lower surface') in leaf_spots
        ;       l_s_position(scattered) in leaf_spots
        ;       l_s_position('upper surface') in leaf_spots
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r53([confirmed(anthracnose)in disorder]) if
        suspected(anthracnose) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(yellow) in leaf_spots,
        (       l_s_position('lower surface') in leaf_spots
        ;       l_s_position(scattered) in leaf_spots
        ;       l_s_position('upper surface') in leaf_spots
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r54([confirmed(gum_spots)in disorder]) if
        suspected(gum_spots) in disorder,
        existence(yes) in leaf_spots,
        l_s_color(brown) in leaf_spots,
        l_s_position('lower surface') in leaf_spots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
super(rules)
}.

verify_disorders :: {
r1([highly_confirmed(rust_mite)in disorder]) if
        confirmed(rust_mite) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_94978) in plant,       :(_94978>=5),
        (       f_color(purple) in fruits
        ;       f_color(rust) in fruits
        ), !,
        f_r_status(rough) in fruits &
r2([highly_confirmed(flat_mite)in disorder]) if
        confirmed(flat_mite) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_96682) in plant,       :(_96682>=5),
        existence(yes) in fruit_spots,
        (       f_s_color(bronze) in fruit_spots
        ;       f_s_color(brown) in fruit_spots
        ;       f_s_color('scabby patches') in fruit_spots
```

```
        ;       f_s_color(sliver) in fruit_spots
        ), !,
        f_s_shape(irregular) in fruit_spots,
        (       f_s_position(rind) in fruit_spots
        ;       f_s_position(scattered) in fruit_spots
        ), ! &
r3([highly_confirmed(flat_mite)in disorder]) if
        confirmed(flat_mite) in disorder,
        season(spring) in plant,
        age(_98942) in plant,       :(_98942>=5),
        fl_status(drop) in flowers &
r4([highly_confirmed(stubborn)in disorder]) if
        confirmed(stubborn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_100202) in plant,      :(_100202>=5),
        f_shape(asymatric) in fruits,
        f_r_status(irregular) in fruits &
r5([highly_confirmed(alternaria_rot)in disorder]) if
        confirmed(alternaria_rot) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_101598) in plant,      :(_101598>=5),
        f_r_status(drop) in fruits &
r6([highly_confirmed(sun_burn)in disorder]) if
        confirmed(sun_burn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_102838) in plant,      :(_102838>=5),
        f_r_status(leathery) in fruits &
r7([highly_confirmed(mediterranean_fruit_fly)in disorder]) if
        confirmed(mediterranean_fruit_fly) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_104082) in plant,      :(_104082>=5),
        (       f_s_shape(circular) in fruit_spots
        ;       f_s_shape('large and circular') in fruit_spots
        ), ! &
r8([highly_confirmed(green_stink_bug)in disorder]) if
        confirmed(green_stink_bug) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_105558) in plant,      :(_105558>=5),
        f_s_shape(irregular) in fruit_spots &
r9([highly_confirmed(leafminer)in disorder]) if
        confirmed(leafminer) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_106798) in plant,      :(_106798>=5),
        f_s_shape('zigzag tunnels') in fruit_spots &
r10([highly_confirmed(impieetratura)in disorder]) if
        confirmed(impieetratura) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
```

```
        ), !,
        age(_108038) in plant,      :(_108038>=5),
        (f_s_shape('gum pocket') in fruit_spots
        ;f_s_shape(raised) in fruit_spots )
        &
r11([highly_confirmed(citrus_flower_moth)in disorder]) if
        confirmed(citrus_flower_moth) in disorder,
        age(_108866) in plant,      :(_108866>=5),
        (       fl_color(brown) in flowers
        ;       fl_color(yellow) in flowers
        ), ! &
r12([highly_confirmed(citrus_flower_moth)in disorder]) if
        confirmed(citrus_flower_moth) in disorder,
        age(_109926) in plant,      :(_109926>=5),
        tw_shape(eaten) in twigs &
r13([highly_confirmed(phosphorus_def)in disorder]) if
        confirmed(phosphorus_def) in disorder,
        season(spring) in plant,
        tw_status(dieback) in twigs &
r14([highly_confirmed(rose_scarab)in disorder]) if
        confirmed(rose_scarab) in disorder,
        season(spring) in plant,
        age(_111643) in plant,      :(_111643>=5),
        fl_status(drop) in flowers &
r15([highly_confirmed(sun_burn)in disorder]) if
        confirmed(sun_burn) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_112903) in plant,      :(_112903>=5),
        f_s_shape(circular) in fruit_spots,
        f_r_status(leathery) in fruits &
r16([highly_confirmed(lichens)in disorder]) if
        confirmed(lichens) in disorder,
        age(_113883) in plant,      :(_113883>=5),
        t_shape('lichen growths') in trunk &
r17([highly_confirmed(ganoderma_rot)in disorder]) if
        confirmed(ganoderma_rot) in disorder,
        age(_114707) in plant,      :(_114707>=5),
        t_shape('fungal growths') in trunk &
r18([highly_confirmed(fruit_cracking)in disorder]) if
        confirmed(fruit_cracking) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_115973) in plant,      :(_115973>=5),
        f_shape(cracks) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r19([highly_confirmed(fruit_creasing)in disorder]) if
        confirmed(fruit_creasing) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_117791) in plant,      :(_117791>=5),
        f_r_status(creasing) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
```

```
        ;       value(valencia) in variety
        ), ! &
r20([highly_confirmed(sooty_mold)in disorder]) if
        confirmed(sooty_mold) in disorder,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), !,
        age(_119609) in plant,      :(_119609>=5),
        f_color(black) in fruits,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r21([highly_confirmed(gummosis)in disorder]) if
        confirmed(gummosis) in disorder,
        age(_121035) in plant,      :(_121035>=5),
        t_shape('gum spots') in trunk,
        (       t_position('basal part') in trunk
        ;       t_position('feeder roots') in trunk
        ), !,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r22([highly_confirmed(psorosis)in disorder]) if
        confirmed(psorosis) in disorder,
        age(_122829) in plant,      :(_122829>=5),
        t_shape('bark scaling') in trunk,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r23([highly_confirmed(wilt_root_rot)in disorder]) if
        confirmed(wilt_root_rot) in disorder,
        (       r_color(black) in roots
        ;       r_color(brown) in roots
        ), ! &
r24([highly_confirmed(rust_mite)in disorder]) if
        confirmed(rust_mite) in disorder,
        b_color(rust) in branches,
        b_type(flushes) in branches &
r25([highly_confirmed(rust_mite)in disorder]) if
        confirmed(rust_mite) in disorder,
        b_status(decline) in branches &
r26([highly_confirmed(brown_mite)in disorder]) if
        confirmed(brown_mite) in disorder,
        l_shape(webbed) in leaves &
r27([highly_confirmed(brown_mite)in disorder]) if
        confirmed(brown_mite) in disorder,
        b_color(pale) in branches &
r28([highly_confirmed(bud_mite)in disorder]) if
        confirmed(bud_mite) in disorder,
        (       b_status(flattened) in branches
        ;       b_status(stunted) in branches
        ;       b_status(thickened) in branches
        ), !,
        b_type(flushes) in branches &
r29([highly_confirmed(bud_mite)in disorder]) if
        confirmed(bud_mite) in disorder,
        f_shape(malformed) in fruits &
```

```
r30([highly_confirmed(citrus_nematude)in disorder]) if
        confirmed(citrus_nematude) in disorder,
        r_color(brown) in roots,
        (       r_status(adhesive) in roots
        ;       r_status(sloughing) in roots
        ), !,
        r_type('feeder roots') in roots &
r31([highly_confirmed(leafminer)in disorder]) if
        confirmed(leafminer) in disorder,
        l_s_shap('zigzag tunnels') in leaf_spots &
r32([highly_confirmed(leafminer)in disorder]) if
        confirmed(leafminer) in disorder,
        l_shape(zigzag_tunnels) in leaves &
r33([highly_confirmed(aphids)in disorder]) if
        confirmed(aphids) in disorder,
        (       i_color(black) in insects
        ;       i_color(green) in insects
        ), !,
        (       i_status(aggregated) in insects
        ;       i_status(stationary) in insects
        ), ! &
r34([highly_confirmed(gummosis)in disorder]) if
        confirmed(gummosis) in disorder,
        b_status('die back') in branches &
r35([highly_confirmed(zinc_def)in disorder]) if
        confirmed(zinc_def) in disorder,
        tw_status(dieback) in twigs &
r36([highly_confirmed(zinc_def)in disorder]) if
        confirmed(zinc_def) in disorder,
        f_shape(small) in fruits &
r37([highly_confirmed(manganese_def)in disorder]) if
        confirmed(manganese_def) in disorder,
        tw_status(dieback) in twigs &
r38([highly_confirmed(potassium_def)in disorder]) if
        confirmed(potassium_def) in disorder,
        tw_status(dieback) in twigs &
r39([highly_confirmed(nitrogen_def)in disorder]) if
        confirmed(nitrogen_def) in disorder,
        l_status(small) in leaves &
r40([highly_confirmed(nitrogen_def)in disorder]) if
        confirmed(nitrogen_def) in disorder,
        tw_status(dieback) in twigs &
r41([highly_confirmed(iron_def)in disorder]) if
        confirmed(iron_def) in disorder,
        f_r_status(reduced) in fruits,
        f_shape(small) in fruits &
r42([highly_confirmed(citrus_white_fly)in disorder]) if
        confirmed(citrus_white_fly) in disorder,
        i_color(white) in insects,
        i_status(flying) in insects &
r43([highly_confirmed(mealy_bug)in disorder]) if
        confirmed(mealy_bug) in disorder,
        i_color(white) in insects,
        i_status(stationary) in insects &
r44([highly_confirmed(anthracnose)in disorder]) if
        confirmed(anthracnose) in disorder,
        b_color(brown) in branches,
        b_status(dry) in branches &
r45([highly_confirmed(anthracnose)in disorder]) if
        confirmed(anthracnose) in disorder,
```

```
        l_s_shap(necrotic) in leaf_spots &
r46([highly_confirmed(alternaria_leaves_spot)in disorder]) if
        confirmed(alternaria_leaves_spot) in disorder,
        l_s_shap('concentric zones') in leaf_spots &
r47([highly_confirmed(scales)in disorder]) if
        confirmed(scales) in disorder,
        i_status(stucked) in insects,
        (       i_color(black) in insects
        ;       i_color(purple) in insects
        ;       i_color(red) in insects
        ;       i_color(white) in insects
        ), ! &
r48([highly_confirmed(scales)in disorder]) if
        confirmed(scales) in disorder,
        (       f_shape(malformed) in fruits
        ;       f_shape(small) in fruits
        ), ! &
r49([highly_confirmed(armillaria_root_rot)in disorder]) if
        confirmed(armillaria_root_rot) in disorder,
        r_status('fungal growth') in roots,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r50([highly_confirmed(sooty_mold)in disorder]) if
        confirmed(sooty_mold) in disorder,
        l_color(black) in leaves,
        l_status(_14079) in leaves,
        :( \+ memberchk(insect_persent, _14079)),
        b_status(_14407) in branches,
        :( \+ memberchk('insect present', _14407)),
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), ! &
r51([highly_confirmed(sooty_mold)in disorder]) if
        confirmed(sooty_mold) in disorder,
        (       value(navel) in variety
        ;       value(succar) in variety
        ;       value(valencia) in variety
        ), !,
        b_color(black) in branches &
r52([highly_confirmed(iron_def)in disorder]) if
        confirmed(iron_def) in disorder,
        ph(_16825) in soil,          :(_16825<8.5),
        ca_carbonate(_17096) in soil,       :(_17096<10),
        iron_def_sp('most trees') in disorder &
r53([highly_confirmed(manganese_def)in disorder]) if
        confirmed(manganese_def) in disorder,
        ph(_17944) in soil,          :(_17944<8.5),
        ca_carbonate(_18215) in soil,       :(_18215<10),
        manganese_def_sp('most trees') in disorder &
r54([highly_confirmed(zinc_def)in disorder]) if
        confirmed(zinc_def) in disorder,
        ph(_19063) in soil,          :(_19063<8.5),
        ca_carbonate(_19334) in soil,       :(_19334<10),
        zinc_def_sp('most trees') in disorder &
r55([highly_confirmed(nitrogen_def)in disorder]) if
        confirmed(nitrogen_def) in disorder,
        water_table_level(_20162) in soil,  :(_20162<1.5),
```

```
        nitrogen_def_sp('most trees') in disorder &
r56([highly_confirmed(salt_injury)in disorder]) if
        confirmed(salt_injury) in disorder,
        ec(_20994) in soil,          :(_20994>=2),
        salt_injury_sp('most trees') in disorder &
r57([highly_confirmed(salt_injury)in disorder]) if
        confirmed(salt_injury) in disorder,
        eciw(_21818) in water,     :(_21818>=1),
        salt_injury_sp('most trees') in disorder &
r58([highly_confirmed(aphids)in disorder]) if
        confirmed(aphids) in disorder,
        season(spring) in plant &
r59([highly_confirmed(citrus_nematude)in disorder]) if
        confirmed(citrus_nematude) in disorder,
        season(spring) in plant &
r60([highly_confirmed(magnesium_def)in disorder]) if
        confirmed(magnesium_def) in disorder,
        eciw(_23744) in water,     :(_23744<1),
        ec(_24007) in soil,          :(_24007<2),
        magnesium_def_sp('most trees') in disorder &
r61([highly_confirmed(calcium_def)in disorder]) if
        confirmed(calcium_def) in disorder,
        eciw(_24851) in water,     :(_24851<1),
        ec(_25114) in soil,          :(_25114<2),
        calcium_def_sp('most trees') in disorder &
r62([highly_confirmed(potassium_def)in disorder]) if
        confirmed(potassium_def) in disorder,
        eciw(_25958) in water,     :(_25958<1),
        ec(_26221) in soil,          :(_26221<2),
        potassium_def_sp('most trees') in disorder &
super(rules)
}.
```

4.2 Inference layer

```
/* File name : diag_inference.pl*/
:- ensure_loaded('$KROL/lib/krol_init').
diag_inference :: {
        input(determine,[] )&
        output(determine,[] )&
        input(predict,[branches-b_color,buds-u_color,flowers-f_l_shape,fruit_spots-existence,fruits-
f_color,fruits-f_r_status,fruits-f_shape,leaf_spots-existence,leaves-l_color,leaves-l_shape,plant-
age,plant-season,roots-r_status,trunk-t_shape,variety-value] )&
output(predict,[disorder-subbected] )&
input(confirm,[branches-b_color,branches-b_ststus,branches-b_type,buds-u_color,buds-u_shape,buds-
u_status,disorder-subbected,flowers-f_l_shape,fruit_spots-existence,fruit_spots-f_s_color,fruit_spots-
f_s_position,fruit_spots-f_s_shape,fruits-f_color,fruits-f_r_status,fruits-f_shape,leaf_spots-
existence,leaf_spots-l_s_color,leaf_spots-l_s_position,leaf_spots-l_s_shap,leaves-l_c_position,leaves-
l_color,leaves-l_shape,leaves-l_status,leaves-l_type,plant-age,plant-season,trunk-t_position,trunk-
t_shape,twigs-tw_color,variety-value] )&
output(confirm,[disorder-confirmed] )&
input(verify,[branches-b_color,branches-b_ststus,branches-b_type,disorder-calcium_def_sp,disorder-
confirmed,disorder-iron_def_sp,disorder-magnesium_def_sp,disorder-manganese_def_sp,disorder-
nitrogen_def_sp,disorder-potassium_def_sp,disorder-salt_injury_sp,disorder-zinc_def_sp,flowers-
fl_color,flowers-fl_status,fruit_spots-existence,fruit_spots-f_s_color,fruit_spots-
f_s_position,fruit_spots-f_s_shape,fruits-f_color,fruits-f_r_status,fruits-f_shape,insects-i_color,insects-
i_status,leaf_spots-l_s_shap,leaves-l_color,leaves-l_shape,leaves-l_status,plant-age,plant-season,roots-
r_color,roots-r_status,roots-r_type,soil-ca_carbonate,soil-ec,soil-ph,soil-water_table_level,trunk-
t_position,trunk-t_shape,twigs-tw_shape,twigs-tw_status,variety-value,water-eciw] )&
        output(verify,[disorder-highly_confirmed] )&
```

```
        description(determine, '') &
        determine :-
                plant_determine_plant :: table &
        description(predict, '') &
        predict :-
                caused_by_disoders :: conclude_all &
        description(confirm, '') &
        confirm :-
                confirm_disorders :: conclude_all &
        description(verify, '') &
        verify :-
                verify_disorders :: conclude_all &
super(krol_init)
}.
```

4.3 Task layer

```
/* File name : diag_task.pl */
task([diag_task]).
% This is to mark that is file is generated by task editor. Please do not delete
diag_task :: {super(krol_init)}.
diag_task_transfer :: {super(diag_task)}.
diag_task_uncondional :: {
start_inference :-
        diag_task_user :: init_inf,
        diag_task_user :: determine_exist,
        diag_task_user :: determine_age,
        diag_task_condional :: plantation_not_exist &

super(diag_task)
}.
diag_task_condional :: {
plantation_not_exist :-
        (       plantation :: get_value(existence(_9824)),
                :(_9824=no) ->
                        diag_task_user :: no_need_for_diag
        ;
                        (diag_inference :: determine,
                        :citex_diag_dlg)
        ) &
super(diag_task)
}.
diag_task_repetitive :: {super(diag_task)}.

diag_task_user :: {
        determine_exist :-
        plantation :: get_value(plantation_date(Pdate)),
        :extract_date(Pdate, Pdate1),
        Pdate1 = [PY, PM, PD, _, _, _],
        :datime(datime(Y,M,D,_,_,_)),
        :(Date = [D, M, Y]),
        :current_week(Date, W),
        plantation :: set(current_date(Date)),
        plant :: set(current_date(Date)),
        plant :: set(current_month(M)),
        plant :: set(current_week(W)),
        (:compare_date(=<, [PD,PM,PY],[D,M,Y]) ->
                plantation :: set(existence(yes))
        ;       plantation :: set(existence(no))
        ) &
```

```
no_need_for_diag :-
krol_msgs :: show("There is no plantation exists to be diagnose ....",[])&
init_inf :-
        krol_init :: init,
        utility :: restart &
determine_age :-
        plantation :: get_value(plantation_date(Pdate)),
        :extract_date(Pdate, Pdate1),
        Pdate1 = [PY, PM, PD, _, _, _],
        :datime(datime(Y,M,D,_,_,_)),
        :dif([PD,PM,PY],[D,M,Y],[_,_,Age]),
        plant :: set(age(Age)) &
super(diag_task)
}.
```

5. Treatment subsystem

5.1 Relations Between Concept

The following code is the relation between concept implementation for treatment subsystem.

```
/* File name : treat_rules.pl */
:- use_module(library(lists), [memberchk/2]).
:- ensure_loaded('$KROL/lib/rule_exp').
treated_by :: {
r1([     material_name(none)in stubborn,
        method(advice)in stubborn,          number(1)in stubborn,
        date(Vv1) in stubborn]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(stubborn) in disorder &
r2([     material_name(none)in stubborn,
        method(advice)in stubborn,          number(1)in stubborn,
        date(Vv1) in stubborn]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(stubborn) in disorder &
r3([     material_name(none)in impieetratura,
        method(advice)in impieetratura,     number(1)in impieetratura,
        date(Vv1) in impieetratura]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(impieetratura) in disorder &
r4([     material_name(none)in impieetratura,method(advice)in impieetratura,
        number(1)in impieetratura,          date(Vv1) in impieetratura]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(impieetratura) in disorder &
r5([     material_name(none)in anthracnose,
        method(advice)in anthracnose,number(1)in anthracnose,
        date(Vv1) in anthracnose]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(anthracnose) in disorder &
r6([     material_name(none)in anthracnose,
        method(advice)in anthracnose,number(1)in anthracnose,
        date(Vv1) in anthracnose]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(anthracnose) in disorder &
r7([     material_name(none)in alternaria_leaves_spot,
        method(advice)in alternaria_leaves_spot,
        number(1)in alternaria_leaves_spot,
        date(Vv1) in alternaria_leaves_spot]) if
        :eval_rule_exp(current_date of plant, Vv1),
```

```
        confirmed(alternaria_leaves_spot) in disorder &
r8([    material_name(none)in alternaria_leaves_spot,
        method(advice)in alternaria_leaves_spot,
        number(1)in alternaria_leaves_spot,
        date(Vv1) in alternaria_leaves_spot]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(alternaria_leaves_spot) in disorder &
r9([    material_name(none)in alternaria_rot,
        method(advice)in alternaria_rot,
        number(1)in alternaria_rot,
        date(Vv1) in alternaria_rot]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(alternaria_rot) in disorder &
r10([   material_name(none)in alternaria_rot,
        method(advice)in alternaria_rot,number(1)in alternaria_rot,
        date(Vv1) in alternaria_rot]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(alternaria_rot) in disorder &
r11([   material_name(none)in gum_spots,
        method(advice)in gum_spots,     number(1)in gum_spots,
        date(Vv1) in gum_spots]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(gum_spots) in disorder &
r12([   material_name(none)in gum_spots,
        method(advice)in gum_spots,     number(1)in gum_spots,
        date(Vv1) in gum_spots]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(gum_spots) in disorder &
r13([   material_name(none)in sun_burn,method(advice)in sun_burn,
        number(1)in sun_burn,date(Vv1) in sun_burn]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(sun_burn) in disorder &
r14([   material_name(none)in sun_burn,method(advice)in sun_burn,
        number(1)in sun_burn,date(Vv1) in sun_burn]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(sun_burn) in disorder &
r15([   material_name(none)in salt_injury, method(advice)in salt_injury,
        number(1)in salt_injury,date(Vv1) in salt_injury]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(salt_injury) in disorder &
r16([   material_name(none)in salt_injury,method(advice)in salt_injury,
        number(1)in salt_injury,date(Vv1) in salt_injury]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(salt_injury) in disorder &
r17([   material_name(none)in rose_scarab,      method(advice)in rose_scarab,
        number(1)in rose_scarab,date(Vv1) in rose_scarab]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(rose_scarab) in disorder &
r18([   material_name(none)in rose_scarab,      method(advice)in rose_scarab,
        number(1)in rose_scarab,date(Vv1) in rose_scarab]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(rose_scarab) in disorder &
r19([   material_name(none)in green_stink_bug,method(advice)in green_stink_bug,
        number(1)in green_stink_bug,date(Vv1) in green_stink_bug]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(green_stink_bug) in disorder &
r20([   material_name(none)in green_stink_bug,method(advice)in green_stink_bug,
        number(1)in green_stink_bug,date(Vv1) in green_stink_bug]) if
        :eval_rule_exp(current_date of plant, Vv1),  highly_confirmed(green_stink_bug) in disorder
&
```

```
r21([    material_name(none)in psorosis,
         method(advice)in psorosis,number(1)in psorosis,
         date(Vv1) in psorosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(psorosis) in disorder &
r22([    material_name(none)in psorosis,method(advice)in psorosis,
         number(1)in psorosis,date(Vv1) in psorosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(psorosis) in disorder &
r23([    material_name(none)in armillaria_root_rot,
         method(advice)in armillaria_root_rot,number(1)in armillaria_root_rot,
         date(Vv1) in armillaria_root_rot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(armillaria_root_rot) in disorder &
r24([    material_name(none)in armillaria_root_rot,
         method(advice)in armillaria_root_rot,number(1)in armillaria_root_rot,
         date(Vv1) in armillaria_root_rot]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(armillaria_root_rot) in disorder &
r25([    material_name(none)in fruit_cracking,
         method(advice)in fruit_cracking,number(1)in fruit_cracking,
         date(Vv1) in fruit_cracking]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(fruit_cracking) in disorder &
r26([    material_name(none)in fruit_cracking,method(advice)in fruit_cracking,
         number(1)in fruit_cracking,date(Vv1) in fruit_cracking]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(fruit_cracking) in disorder &
r27([    material_name(none)in fruit_creasing,method(advice)in fruit_creasing,
         number(1)in fruit_creasing,date(Vv1) in fruit_creasing]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(fruit_creasing) in disorder &
r28([    material_name(none)in fruit_creasing,method(advice)in fruit_creasing,
         number(1)in fruit_creasing,date(Vv1) in fruit_creasing]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(fruit_creasing) in disorder &
r29([    material_name(none)in sooty_mold,method(advice)in sooty_mold,
         number(1)in sooty_mold,date(Vv1) in sooty_mold]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(sooty_mold) in disorder &
r30([    material_name(none)in sooty_mold,          method(advice)in sooty_mold,
         number(1)in sooty_mold,date(Vv1) in sooty_mold]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(sooty_mold) in disorder &
r31([    material_name(potassium_permenganat)in gummosis,
         method(disinfection)in gummosis,number(1)in gummosis,
         date(Vv1) in gummosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(gummosis) in disorder &
r32([    material_name(potassium_permenganat)in gummosis,
         method(disinfection)in gummosis,number(1)in gummosis,
         date(Vv1) in gummosis]) if
         :eval_rule_exp(current_date of plant, Vv1),
         highly_confirmed(gummosis) in disorder &
r33a([   material_name(topsin)in ganoderma_rot_op1,
         method('chemical spray')in ganoderma_rot_op1,
         number(1)in ganoderma_rot_op1,date(Vv1) in ganoderma_rot_op1]) if
         :eval_rule_exp(current_date of plant, Vv1),
         confirmed(ganoderma_rot) in disorder &
r33b([   method('chemical spray')in ganoderma_rot_op2,
```

```
        material_name('bordeaux past')in ganoderma_rot_op2,
        number(2)in ganoderma_rot_op2,date(Vv1) in ganoderma_rot_op2]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(ganoderma_rot) in disorder &
r34a([  material_name(topsin)in ganoderma_rot_op1,
        method('chemical spray')in ganoderma_rot_op1,
        number(1)in ganoderma_rot_op1,date(Vv1) in ganoderma_rot_op1]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(ganoderma_rot) in disorder &
r34b([  method('chemical spray')in ganoderma_rot_op2,
        material_name('bordeaux past')in ganoderma_rot_op2,
        number(2)in ganoderma_rot_op2,date(Vv1) in ganoderma_rot_op2]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(ganoderma_rot) in disorder &
r35a([  material_name(topsin)in wilt_root_rot_op1,
        method('soil treatment')in wilt_root_rot_op1,
        number(1)in wilt_root_rot_op1,date(Vv1) in wilt_root_rot_op1]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(wilt_root_rot) in disorder &
r35b([  material_name(topsin)in wilt_root_rot_op2,
        method('soil treatment')in wilt_root_rot_op2,
        number(2)in wilt_root_rot_op2,date(Vv1) in wilt_root_rot_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv1),
        confirmed(wilt_root_rot) in disorder &
r36a([  material_name(topsin)in wilt_root_rot_op1,
        method('soil treatment')in wilt_root_rot_op1,
        number(1)in wilt_root_rot_op1,date(Vv1) in wilt_root_rot_op1]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(wilt_root_rot) in disorder &
r36b([  material_name(topsin)in wilt_root_rot_op2,
        method('soil treatment')in wilt_root_rot_op2,
        number(2)in wilt_root_rot_op2,date(Vv1) in wilt_root_rot_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv1),
        highly_confirmed(wilt_root_rot) in disorder &
r37([   material_name('vertimec 1.8%')in citrus_white_fly,
        method('chemical spray')in citrus_white_fly,number(1)in citrus_white_fly,
        date(Vv1) in citrus_white_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(citrus_white_fly) in disorder,confirmed(_55189) in disorder,
        :( \+ memberchk(aphids, _55189)) &
r38([   material_name('vertimec 1.8%')in citrus_white_fly,
        method('chemical spray')in citrus_white_fly,
        number(1)in citrus_white_fly,date(Vv1) in citrus_white_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(citrus_white_fly) in disorder,
        highly_confirmed(_56825) in disorder,
        :( \+ memberchk(aphids, _56825)) &
r39([   method('chemical spray')in aphids,
        method('chemical spray')in citrus_white_fly,
        number(1)in aphids,         number(1)in citrus_white_fly,
        date(Vv1) in aphids,date(Vv1) in citrus_white_fly,
        material_name(Vv2) in aphids,material_name(Vv2) in citrus_white_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),confirmed(aphids) in disorder,
        confirmed(citrus_white_fly) in disorder,material_gr1(Vv2) in operation &
r40([   method('chemical spray')in aphids,
        method('chemical spray')in citrus_white_fly,number(1)in aphids,
        number(1)in citrus_white_fly,material_name(Vv1) in aphids,
        material_name(Vv1) in citrus_white_fly,date(Vv2) in aphids,
        date(Vv2) in citrus_white_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
```

```
        highly_confirmed(aphids) in disorder,          highly_confirmed(citrus_white_fly) in disorder,
        material_gr1(Vv1) in operation &
r41([   material_name('malathion 57%')in aphids,
        method('chemical spray')in aphids,  number(1)in aphids,
        date(Vv1) in aphids]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(aphids) in disorder,confirmed(_62849) in disorder,
        :( \+ memberchk(citrus_white_fly, _62849)) &
r42([   material_name('malathion 57%')in aphids,
        method('chemical spray')in aphids,number(1)in aphids,
        date(Vv1) in aphids]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(aphids) in disorder,highly_confirmed(_64485) in disorder,
        :( \+ memberchk(citrus_white_fly, _64485)) &
r43([   method('chemical spray')in citrus_flower_moth,
        number(1)in citrus_flower_moth,material_name(Vv1) in citrus_flower_moth,
        date(Vv2) in citrus_flower_moth]) if
        :eval_rule_exp(current_date of plant, Vv2),
        confirmed(citrus_flower_moth) in disorder,material_gr2(Vv1) in operation &
r44([   method('chemical spray')in citrus_flower_moth,
        number(1)in citrus_flower_moth,material_name(Vv1) in citrus_flower_moth,
        date(Vv2) in citrus_flower_moth]) if
        :eval_rule_exp(current_date of plant, Vv2),
        highly_confirmed(citrus_flower_moth) in disorder,
        material_gr2(Vv1) in operation &
r45([   method('chemical spray')in lichens,number(1)in lichens,
        material_name(Vv1) in lichens,date(Vv2) in lichens]) if
        :eval_rule_exp(current_date of plant, Vv2),season(winter) in plant,
        confirmed(lichens) in disorder,material_gr3(Vv1) in operation &
r46([   method('chemical spray')in lichens,number(1)in lichens,
        material_name(Vv1) in lichens,date(Vv2) in lichens]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(winter) in plant,highly_confirmed(lichens) in disorder,
        material_gr3(Vv1) in operation &
r47([   method('chemical spray')in lichens,number(1)in lichens,
        special_date('next 1/12')in lichens,material_name(Vv1) in lichens]) if
        confirmed(lichens) in disorder,season(_72183) in plant,          :(_72183\==winter),
        material_gr3(Vv1) in operation &
r48([   method('chemical spray')in lichens,number(1)in lichens,
        special_date('next 1/12')in lichens,
        material_name(Vv1) in lichens]) if
        season(_73566) in plant,    :(_73566\==winter),
        highly_confirmed(lichens) in disorder,material_gr3(Vv1) in operation &
r49([   material_name('bordeaux past')in gummosis,method(painting)in gummosis,
        number(1)in gummosis,date(Vv1) in gummosis]) if
        :eval_rule_exp(current_date of plant, Vv1),season(winter) in plant,
        confirmed(gummosis) in disorder &
r50([   material_name('bordeaux past')in gummosis,
        method(painting)in gummosis,number(1)in gummosis,
        date(Vv1) in gummosis]) if
        :eval_rule_exp(current_date of plant, Vv1),  season(winter) in plant,
        highly_confirmed(gummosis) in disorder &
r51([   method('chemical spray')in scales,number(1)in scales,
        material_name(Vv1) in scales,date(Vv2) in scales]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,confirmed(scales) in disorder,
        material_gr1(Vv1) in operation &
r52([   method('chemical spray')in scales,number(1)in scales,
        material_name(Vv1) in scales,date(Vv2) in scales]) if
        :eval_rule_exp(current_date of plant, Vv2),season(summer) in plant,
```

```
        highly_confirmed(scales) in disorder,material_gr1(Vv1) in operation &
r53([   method('chemical spray')in mealy_bug,number(1)in mealy_bug,
        material_name(Vv1) in mealy_bug,date(Vv2) in mealy_bug]) if
        :eval_rule_exp(current_date of plant, Vv2),  season(summer) in plant,
        confirmed(mealy_bug) in disorder,material_gr1(Vv1) in operation &
r54([   method('chemical spray')in mealy_bug,number(1)in mealy_bug,
        material_name(Vv1) in mealy_bug,date(Vv2) in mealy_bug]) if
        :eval_rule_exp(current_date of plant, Vv2),season(summer) in plant,
        highly_confirmed(mealy_bug) in disorder,
        material_gr1(Vv1) in operation &
r55([   method('chemical spray')in scales,   number(1)in scales,
        special_date('next summer')in scales,          material_name(Vv1) in scales]) if
        season(spring) in plant,confirmed(scales) in disorder,
        material_gr1(Vv1) in operation &
r56([   method('chemical spray')in scales,   number(1)in scales,
        special_date('next summer')in scales,          material_name(Vv1) in scales]) if
        season(spring) in plant,highly_confirmed(scales) in disorder,
        material_gr1(Vv1) in operation &
r57([   method('chemical spray')in mealy_bug,number(1)in mealy_bug,
        special_date('next summer')in mealy_bug,
        material_name(Vv1) in mealy_bug]) if
        season(spring) in plant,confirmed(mealy_bug) in disorder,
        material_gr1(Vv1) in operation &
r58([   method('chemical spray')in mealy_bug,number(1)in mealy_bug,
        special_date('next summer')in mealy_bug,
        material_name(Vv1) in mealy_bug]) if
        season(spring) in plant,highly_confirmed(mealy_bug) in disorder,
        material_gr1(Vv1) in operation &
r59([   method('chemical spray')in scales,number(1)in scales,
        material_name(Vv1) in scales,date(Vv2) in scales]) if
        :eval_rule_exp(current_date of plant, Vv2),  season(winter) in plant,
        confirmed(scales) in disorder,        material_gr4(Vv1) in operation &
r60([   method('chemical spray')in scales,number(1)in scales,
        material_name(Vv1) in scales,date(Vv2) in scales]) if
        :eval_rule_exp(current_date of plant, Vv2),season(winter) in plant,
        highly_confirmed(scales) in disorder,material_gr4(Vv1) in operation &
r61([   method('chemical spray')in mealy_bug,number(1)in mealy_bug,
        material_name(Vv1) in mealy_bug,date(Vv2) in mealy_bug]) if
        :eval_rule_exp(current_date of plant, Vv2),  season(winter) in plant,
        confirmed(mealy_bug) in disorder, material_gr4(Vv1) in operation &
r62([   method('chemical spray')in mealy_bug,
        number(1)in mealy_bug,material_name(Vv1) in mealy_bug,
        date(Vv2) in mealy_bug]) if
        :eval_rule_exp(current_date of plant, Vv2),  season(winter) in plant,
        highly_confirmed(mealy_bug) in disorder,material_gr4(Vv1) in operation &
r63([   method('chemical spray')in scales,   number(1)in scales,
        special_date('next winter')in scales, material_name(Vv1) in scales]) if
        season(autumn) in plant,confirmed(scales) in disorder,
        material_gr4(Vv1) in operation &
r64([   method('chemical spray')in scales,   number(1)in scales,
        special_date('next winter')in scales, material_name(Vv1) in scales]) if
        season(autumn) in plant,highly_confirmed(scales) in disorder,
        material_gr4(Vv1) in operation &
r65([   method('chemical spray')in mealy_bug,number(1)in mealy_bug,
        special_date('next winter')in mealy_bug,
        material_name(Vv1) in mealy_bug]) if
        season(autumn) in plant,confirmed(mealy_bug) in disorder,
        material_gr4(Vv1) in operation &
r66([   method('chemical spray')in mealy_bug,number(1)in mealy_bug,
        special_date('next winter')in mealy_bug,
```

```
        material_name(Vv1) in mealy_bug]) if
        season(autumn) in plant,highly_confirmed(mealy_bug) in disorder,
        material_gr4(Vv1) in operation &
r67a([  method('chemical spray')in leafminer_op1,
        number(1)in leafminer_op1,          material_name(Vv1) in leafminer_op1,
        date(Vv2) in leafminer_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,confirmed(leafminer) in disorder,
        material_gr5(Vv1) in operation &
r67b([  method('chemical spray')in leafminer_op2,number(2)in leafminer_op2,
        material_name(Vv1) in leafminer_op2,date(Vv2) in leafminer_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv2),
        season(summer) in plant,confirmed(leafminer) in disorder,
        material_gr5(Vv1) in operation &
r67c([  method('chemical spray')in leafminer_op3,number(3)in leafminer_op3,
        material_name(Vv1) in leafminer_op3,date(Vv2) in leafminer_op3]) if
        :eval_rule_exp(current_date of plant+42, Vv2),season(summer) in plant,
        confirmed(leafminer) in disorder,material_gr5(Vv1) in operation &
r68a([  method('chemical spray')in leafminer_op1,number(1)in leafminer_op1,
        material_name(Vv1) in leafminer_op1,date(Vv2) in leafminer_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),season(summer) in plant,
        highly_confirmed(leafminer) in disorder,material_gr5(Vv1) in operation &
r68b([  method('chemical spray')in leafminer_op2,number(2)in leafminer_op2,
        material_name(Vv1) in leafminer_op2,date(Vv2) in leafminer_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv2),season(summer) in plant,
        highly_confirmed(leafminer) in disorder,material_gr5(Vv1) in operation &
r68c([  method('chemical spray')in leafminer_op3,number(3)in leafminer_op3,
        material_name(Vv1) in leafminer_op3,date(Vv2) in leafminer_op3]) if
        :eval_rule_exp(current_date of plant+42, Vv2),season(summer) in plant,
        highly_confirmed(leafminer) in disorder,material_gr5(Vv1) in operation &
r69a([  method('chemical spray')in leafminer_op1,number(1)in leafminer_op1,
        special_date('next 1/6')in leafminer_op1,
        material_name(Vv1) in leafminer_op1]) if
        confirmed(leafminer) in disorder,season(_112909) in plant,
        :(_112909\==summer),material_gr5(Vv1) in operation &
r69b([  method('chemical spray')in leafminer_op2,   number(2)in leafminer_op2,
        special_date('next 22/6')in leafminer_op2,
        material_name(Vv1) in leafminer_op2]) if
        confirmed(leafminer) in disorder,season(_114448) in plant,
        :(_114448\==summer),material_gr5(Vv1) in operation &
r69c([  method('chemical spray')in leafminer_op3,number(3)in leafminer_op3,
        special_date('next 13/7')in leafminer_op3,
        material_name(Vv1) in leafminer_op3]) if
        confirmed(leafminer) in disorder,
        season(_115987) in plant, :(_115987\==summer),
        material_gr5(Vv1) in operation &
r70a([  method('chemical spray')in leafminer_op1,
        number(1)in leafminer_op1,          special_date('next 1/6')in leafminer_op1,
        material_name(Vv1) in leafminer_op1]) if
        highly_confirmed(leafminer) in disorder,
        season(_119065) in plant, :(_119065\==summer),
        material_gr5(Vv1) in operation &
r70b([  method('chemical spray')in leafminer_op2,
        number(1)in leafminer_op2,          special_date('next 22/6')in leafminer_op2,
        material_name(Vv1) in leafminer_op2]) if
        highly_confirmed(leafminer) in disorder,
        season(_120604) in plant, :(_120604\==summer),
        material_gr5(Vv1) in operation &
r70c([  method('chemical spray')in leafminer_op3,
        number(1)in leafminer_op3,          special_date('next 13/7')in leafminer_op3,
```

```
        material_name(Vv1) in leafminer_op3]) if
        highly_confirmed(leafminer) in disorder,
        season(_122143) in plant, :(_122143\==summer),
        material_gr5(Vv1) in operation &
r71a([  method('chemical spray')in rust_mite_op1,
        number(1)in rust_mite_op1,        material_name(Vv1) in rust_mite_op1,
        date(Vv2) in rust_mite_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,
        confirmed(rust_mite) in disorder,  material_gr6(Vv1) in operation &
r71b([  method('chemical spray')in rust_mite_op2,
        number(2)in rust_mite_op2,        material_name(Vv1) in rust_mite_op2,
        date(Vv2) in rust_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        season(summer) in plant,  confirmed(rust_mite) in disorder,
        material_gr6(Vv1) in operation &
r72a([  method('chemical spray')in rust_mite_op1,
        number(1)in rust_mite_op1,        material_name(Vv1) in rust_mite_op1,
        date(Vv2) in rust_mite_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,  highly_confirmed(rust_mite) in disorder,
        material_gr6(Vv1) in operation &
r72b([  method('chemical spray')in rust_mite_op2,
        number(2)in rust_mite_op2,        material_name(Vv1) in rust_mite_op2,
        date(Vv2) in rust_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        season(summer) in plant,  highly_confirmed(rust_mite) in disorder,
        material_gr6(Vv1) in operation &
r73([   material_name(none)in rust_mite,method(advice)in rust_mite,
        number(1)in rust_mite,    date(Vv1) in rust_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_130378) in plant, :(_130378\==summer),
        confirmed(rust_mite) in disorder &
r74([   material_name(none)in rust_mite,  method(advice)in rust_mite,
        number(1)in rust_mite,    date(Vv1) in rust_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_131949) in plant, :(_131949\==summer),
        highly_confirmed(rust_mite) in disorder &
r75a([  method('chemical spray')in bud_mite_op1,
        number(1)in bud_mite_op1,         material_name(Vv1) in bud_mite_op1,
        date(Vv2) in bud_mite_op1]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        confirmed(bud_mite) in disorder,
        current_week(_133738) in plant,   :(_133738>=7),
        current_week(_134001) in plant,   :(_134001=<22),
        material_gr6(Vv1) in operation &
r75b([  method('chemical spray')in bud_mite_op2,
        number(2)in bud_mite_op2,         material_name(Vv1) in bud_mite_op2,
        date(Vv2) in bud_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        confirmed(bud_mite) in disorder,current_week(_135790) in plant,
        :(_135790>=7),current_week(_136053) in plant,      :(_136053=<22),
        material_gr6(Vv1) in operation &
r75c([  method('chemical spray')in bud_mite_op1,
        number(1)in bud_mite_op1,         material_name(Vv1) in bud_mite_op1,
        date(Vv2) in bud_mite_op1]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        confirmed(bud_mite) in disorder,
        current_week(_137842) in plant,   :(_137842>=35),
        current_week(_138105) in plant,   :(_138105=<44),
```

```
        material_gr6(Vv1) in operation &
r75d([  method('chemical spray')in bud_mite_op2,
        number(2)in bud_mite_op2,          material_name(Vv1) in bud_mite_op2,
        date(Vv2) in bud_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        confirmed(bud_mite) in disorder,   current_week(_139894) in plant,
        :(_139894>=35),current_week(_140157) in plant,     :(_140157=<44),
        material_gr6(Vv1) in operation &
r76a([  method('chemical spray')in bud_mite_op1,
        number(1)in bud_mite_op1,          material_name(Vv1) in bud_mite_op1,
        date(Vv2) in bud_mite_op1]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        highly_confirmed(bud_mite) in disorder,
        current_week(_141946) in plant,    :(_141946>=7),
        current_week(_142209) in plant,    :(_142209=<22),
        material_gr6(Vv1) in operation &
r76b([  method('chemical spray')in bud_mite_op1,
        number(1)in bud_mite_op1,          material_name(Vv1) in bud_mite_op1,
        date(Vv2) in bud_mite_op1]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        highly_confirmed(bud_mite) in disorder,
        current_week(_144258) in plant,    :(_144258>=35),
        current_week(_144521) in plant,    :(_144521=<44),
        material_gr6(Vv1) in operation &
r76c([  method('chemical spray')in bud_mite_op2,
        number(2)in bud_mite_op2,          material_name(Vv1) in bud_mite_op2,
        date(Vv2) in bud_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        highly_confirmed(bud_mite) in disorder,
        current_week(_146310) in plant,    :(_146310>=7),
        current_week(_146573) in plant,    :(_146573=<22),
        material_gr6(Vv1) in operation &
r76d([  method('chemical spray')in bud_mite_op2,
        number(2)in bud_mite_op2,          material_name(Vv1) in bud_mite_op2,
        date(Vv2) in bud_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        highly_confirmed(bud_mite) in disorder,
        current_week(_148362) in plant,    :(_148362>=35),
        current_week(_148625) in plant,    :(_148625=<44),
        material_gr6(Vv1) in operation &
r77a([  material_name(none)in bud_mite,
        method(advice)in bud_mite,         number(1)in bud_mite,
        date(Vv1) in bud_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(bud_mite) in disorder,   current_week(_150378) in plant,    :(_150378>0),
        current_week(_150641) in plant,    :(_150641<7) &
r77b([  material_name(none)in bud_mite,method(advice)in bud_mite,
        number(1)in bud_mite,date(Vv1) in bud_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(bud_mite) in disorder,
        current_week(_152238) in plant,    :(_152238>22),
        current_week(_152501) in plant,    :(_152501<35) &
r77c([  material_name(none)in bud_mite,method(advice)in bud_mite,
        number(1)in bud_mite,date(Vv1) in bud_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        confirmed(bud_mite) in disorder,current_week(_154098) in plant,
        :(_154098>44),current_week(_154361) in plant,      :(_154361=<52) &
r78a([  material_name(none)in bud_mite,method(advice)in bud_mite,
        number(1)in bud_mite,     date(Vv1) in bud_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
```

```
        highly_confirmed(bud_mite) in disorder,
        current_week(_155958) in plant,    :(_155958>0),
        current_week(_156221) in plant,    :(_156221<7) &
r78b([  material_name(none)in bud_mite,   method(advice)in bud_mite,
        number(1)in bud_mite,     date(Vv1) in bud_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(bud_mite) in disorder,
        current_week(_157818) in plant,    :(_157818>22),
        current_week(_158081) in plant,    :(_158081<35) &
r78c([  material_name(none)in bud_mite,method(advice)in bud_mite,
        number(1)in bud_mite,date(Vv1) in bud_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        highly_confirmed(bud_mite) in disorder,
        current_week(_159678) in plant,    :(_159678>44),
        current_week(_159941) in plant,    :(_159941=<52) &
r79a([  method('chemical spray')in brown_mite_op1,
        number(1)in brown_mite_op1,
        material_name(Vv1) in brown_mite_op1,
        date(Vv2) in brown_mite_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,confirmed(brown_mite) in disorder,
        material_gr7(Vv1) in operation &
r79b([  method('chemical spray')in brown_mite_op2,
        number(2)in brown_mite_op2,material_name(Vv1) in brown_mite_op2,
        date(Vv2) in brown_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        season(summer) in plant,confirmed(brown_mite) in disorder,
        material_gr7(Vv1) in operation &
r80a([  method('chemical spray')in brown_mite_op1,
        number(1)in brown_mite_op1,
        material_name(Vv1) in brown_mite_op1,
        date(Vv2) in brown_mite_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,highly_confirmed(brown_mite) in disorder,
        material_gr7(Vv1) in operation &
r80b([  method('chemical spray')in brown_mite_op2,
        number(2)in brown_mite_op2,
        material_name(Vv1) in brown_mite_op2,
        date(Vv2) in brown_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        season(summer) in plant,highly_confirmed(brown_mite) in disorder,
        material_gr7(Vv1) in operation &
r81([   material_name(none)in brown_mite,method(advice)in brown_mite,
        number(1)in brown_mite,date(Vv1) in brown_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),  season(_167980) in plant,
        :(_167980\==summer),confirmed(brown_mite) in disorder &
r82([   material_name(none)in brown_mite,method(advice)in brown_mite,
        number(1)in brown_mite,date(Vv1) in brown_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_169551) in plant,  :(_169551\==summer),
        highly_confirmed(brown_mite) in disorder &
r83a([  method('chemical spray')in flat_mite_op1,
        number(1)in flat_mite_op1,          material_name(Vv1) in flat_mite_op1,
        date(Vv2) in flat_mite_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,confirmed(flat_mite) in disorder,
        material_gr7(Vv1) in operation &
r83b([  method('chemical spray')in flat_mite_op2,
        number(2)in flat_mite_op2,          material_name(Vv1) in flat_mite_op2,
        date(Vv2) in flat_mite_op2]) if
```

```
        :eval_rule_exp(current_date of plant+15, Vv2),
        season(summer) in plant,   confirmed(flat_mite) in disorder,
        material_gr7(Vv1) in operation &
r84a([  method('chemical spray')in flat_mite_op1,
        number(1)in flat_mite_op1,
        material_name(Vv1) in flat_mite_op1,
        date(Vv2) in flat_mite_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(summer) in plant,
        highly_confirmed(flat_mite) in disorder,
        material_gr7(Vv1) in operation &
r84b([  method('chemical spray')in flat_mite_op2,
        number(2)in flat_mite_op2,
        material_name(Vv1) in flat_mite_op2,
        date(Vv2) in flat_mite_op2]) if
        :eval_rule_exp(current_date of plant+15, Vv2),
        season(summer) in plant,
        highly_confirmed(flat_mite) in disorder,
        material_gr7(Vv1) in operation &
r85([   material_name(none)in flat_mite,
        method(advice)in flat_mite,
        number(1)in flat_mite,
        date(Vv1) in flat_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_177746) in plant,  :(_177746\==summer),
        confirmed(flat_mite) in disorder &
r86([   material_name(none)in flat_mite,
        method(advice)in flat_mite,
        number(1)in flat_mite,
        date(Vv1) in flat_mite]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_179317) in plant,  :(_179317\==summer),
        highly_confirmed(flat_mite) in disorder &
r87a1([ method('soil treatment')in citrus_nematude_op1,
        number(1)in citrus_nematude_op1,
        material_name(Vv1) in citrus_nematude_op1,
        date(Vv2) in citrus_nematude_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        confirmed(citrus_nematude) in disorder,
        current_month(2) in plant,
        material_gr8(Vv1) in operation &
r87a2([ method('soil treatment')in citrus_nematude_op2,
        number(2)in citrus_nematude_op2,
        material_name(Vv1) in citrus_nematude_op2,
        date(Vv2) in citrus_nematude_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv2),
        confirmed(citrus_nematude) in disorder,
        current_month(2) in plant,
        material_gr8(Vv1) in operation &
r87b1([ method('soil treatment')in citrus_nematude_op1,
        number(1)in citrus_nematude_op1,
        material_name(Vv1) in citrus_nematude_op1,
        date(Vv2) in citrus_nematude_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        confirmed(citrus_nematude) in disorder,
        current_month(3) in plant,
        material_gr8(Vv1) in operation &
r87b2([ method('soil treatment')in citrus_nematude_op2,
        number(2)in citrus_nematude_op2,
        material_name(Vv1) in citrus_nematude_op2,
```

```
        date(Vv2) in citrus_nematude_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv2),
        confirmed(citrus_nematude) in disorder,
        current_month(3) in plant,
        material_gr8(Vv1) in operation &
r88a1([ method('soil treatment')in citrus_nematude_op1,
        number(1)in citrus_nematude_op1,
        material_name(Vv1) in citrus_nematude_op1,
        date(Vv2) in citrus_nematude_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        highly_confirmed(citrus_nematude) in disorder,
        current_month(2) in plant,
        material_gr8(Vv1) in operation &
r88a2([ method('soil treatment')in citrus_nematude_op2,
        number(2)in citrus_nematude_op2,
        material_name(Vv1) in citrus_nematude_op2,
        date(Vv2) in citrus_nematude_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv2),
        highly_confirmed(citrus_nematude) in disorder,
        current_month(2) in plant,
        material_gr8(Vv1) in operation &
r88b1([ method('soil treatment')in citrus_nematude_op1,
        number(1)in citrus_nematude_op1,
        material_name(Vv1) in citrus_nematude_op1,
        date(Vv2) in citrus_nematude_op1]) if
        :eval_rule_exp(current_date of plant, Vv2),
        highly_confirmed(citrus_nematude) in disorder,
        current_month(3) in plant,
        material_gr8(Vv1) in operation &
r88b2([ method('soil treatment')in citrus_nematude_op2,
        number(2)in citrus_nematude_op2,
        material_name(Vv1) in citrus_nematude_op2,
        date(Vv2) in citrus_nematude_op2]) if
        :eval_rule_exp(current_date of plant+21, Vv2),
        highly_confirmed(citrus_nematude) in disorder,
        current_month(3) in plant,
        material_gr8(Vv1) in operation &
r89a1([ method('soil treatment')in citrus_nematude_op1,
        number(1)in citrus_nematude_op1,
        special_date('next 1/2')in citrus_nematude_op1,
        material_name(Vv1) in citrus_nematude_op1]) if
        confirmed(citrus_nematude) in disorder,
        current_month(_187480) in plant,   :(_187480\==2),
        material_gr8(Vv1) in operation &
r89a2([ method('soil treatment')in citrus_nematude_op2,
        number(2)in citrus_nematude_op2,
        special_date('next 22/2')in citrus_nematude_op2,
        material_name(Vv1) in citrus_nematude_op2]) if
        confirmed(citrus_nematude) in disorder,
        current_month(_187480) in plant,   :(_187480\==2),
        material_gr8(Vv1) in operation &
r89b1([ method('soil treatment')in citrus_nematude_op1,
        number(1)in citrus_nematude_op1,
        special_date('next 1/2')in citrus_nematude_op1,
        material_name(Vv1) in citrus_nematude_op1]) if
        confirmed(citrus_nematude) in disorder,
        current_month(_189019) in plant,   :(_189019\==3),
        material_gr8(Vv1) in operation &
r89b2([ method('soil treatment')in citrus_nematude_op2,
        number(2)in citrus_nematude_op2,
```

```
            special_date('next 22/2')in citrus_nematude_op2,
            material_name(Vv1) in citrus_nematude_op2]) if
            confirmed(citrus_nematude) in disorder,
            current_month(_189019) in plant,   :(_189019\==3),
            material_gr8(Vv1) in operation &
r90a1([     method('soil treatment')in citrus_nematude_op1,
            number(1)in citrus_nematude_op1,
            special_date('next 1/2')in citrus_nematude_op1,
            material_name(Vv1) in citrus_nematude_op1]) if
            highly_confirmed(citrus_nematude) in disorder,
            current_month(_190558) in plant,   :(_190558\==2),
            material_gr8(Vv1) in operation &
r90a2([     method('soil treatment')in citrus_nematude_op2,
            number(2)in citrus_nematude_op2,
            special_date('next 22/2')in citrus_nematude_op2,
            material_name(Vv1) in citrus_nematude_op2]) if
            highly_confirmed(citrus_nematude) in disorder,
            current_month(_190558) in plant,   :(_190558\==2),
            material_gr8(Vv1) in operation &
r90b1([     method('soil treatment')in citrus_nematude_op1,
            number(1)in citrus_nematude_op1,
            special_date('next 1/2')in citrus_nematude_op1,
            material_name(Vv1) in citrus_nematude_op1]) if
            highly_confirmed(citrus_nematude) in disorder,
            current_month(_192097) in plant,   :(_192097\==3),
            material_gr8(Vv1) in operation &
r90b2([     method('soil treatment')in citrus_nematude_op2,
            number(2)in citrus_nematude_op2,
            special_date('next 22/2')in citrus_nematude_op2,
            material_name(Vv1) in citrus_nematude_op2]) if
            highly_confirmed(citrus_nematude) in disorder,
            current_month(_192097) in plant,   :(_192097\==3),
            material_gr8(Vv1) in operation &
r91([       method('foliage nutrition')in nitrogen_def,
            number(1)in nitrogen_def,
            material_name(Vv1) in nitrogen_def,
            date(Vv2) in nitrogen_def]) if
            :eval_rule_exp(current_date of plant, Vv2),
            season(_193704) in plant,  :(_193704\==winter),
            confirmed(nitrogen_def) in disorder,
            material_gr9(Vv1) in operation &
r92([       method('foliage nutrition')in nitrogen_def,
            number(1)in nitrogen_def,
            material_name(Vv1) in nitrogen_def,
            date(Vv1) in nitrogen_def]) if
            :eval_rule_exp(current_date of plant, Vv1),
            season(_195467) in plant,  :(_195467\==winter),
            highly_confirmed(nitrogen_def) in disorder,
            material_gr9(Vv1) in operation &
r93([       material_name('triple phosphate')in phosphorus_def,
            method('foliage nutrition')in phosphorus_def,
            number(1)in phosphorus_def,
            date(Vv1) in phosphorus_def]) if
            :eval_rule_exp(current_date of plant, Vv1),
            season(_197194) in plant,  :(_197194\==winter),
            confirmed(phosphorus_def) in disorder &
r94([       material_name('triple phosphate')in phosphorus_def,
            method('foliage nutrition')in phosphorus_def,
            number(1)in phosphorus_def,
            date(Vv1) in phosphorus_def]) if
```

```
        :eval_rule_exp(current_date of plant, Vv1),
        season(_198765) in plant, :(_198765\==winter),
        highly_confirmed(phosphorus_def) in disorder &
r95([   method('foliage nutrition')in potassium_def,
        number(1)in potassium_def,
        material_name(Vv1) in potassium_def,
        date(Vv2) in potassium_def]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(_200372) in plant, :(_200372\==winter),
        confirmed(potassium_def) in disorder,
        material_gr10(Vv1) in operation &
r96([   method('foliage nutrition')in potassium_def,
        number(1)in potassium_def,
        material_name(Vv1) in potassium_def,
        date(Vv2) in potassium_def]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(_202135) in plant, :(_202135\==winter),
        highly_confirmed(potassium_def) in disorder,
        material_gr10(Vv1) in operation &
r97([   material_name(magnesium_sulfate)in magnesium_def,
        method('foliage nutrition')in magnesium_def,
        number(1)in magnesium_def,
        date(Vv1) in magnesium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
        confirmed(magnesium_def) in disorder &
r98([   material_name(magnesium_sulfate)in magnesium_def,
        method('foliage nutrition')in magnesium_def,
        number(1)in magnesium_def,
        date(Vv1) in magnesium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
        highly_confirmed(magnesium_def) in disorder &
r99([   material_name('micro element mixture')in manganese_def,
        method('foliage nutrition')in manganese_def,
        number(1)in manganese_def,
        date(Vv1) in manganese_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
        confirmed(manganese_def) in disorder &
r100([  material_name('micro element mixture')in manganese_def,
        method('foliage nutrition')in manganese_def,
        number(1)in manganese_def,
        date(Vv1) in manganese_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
        highly_confirmed(manganese_def) in disorder &
r101([  material_name('micro element mixture')in iron_def,
        method('foliage nutrition')in iron_def,
        number(1)in iron_def,
        date(Vv1) in iron_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
        confirmed(iron_def) in disorder &
r102([  material_name('micro element mixture')in iron_def,
        method('foliage nutrition')in iron_def,
        number(1)in iron_def,
        date(Vv1) in iron_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
```

```
        highly_confirmed(iron_def) in disorder &
r103([  method('foliage nutrition')in calcium_def,
        number(1)in calcium_def,
        material_name(Vv1) in calcium_def,
        date(Vv2) in calcium_def]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(_212682) in plant,  :(_212682\==winter),
        confirmed(calcium_def) in disorder,
        material_gr11(Vv1) in operation &
r104([  method('foliage nutrition')in calcium_def,
        number(1)in calcium_def,
        material_name(Vv1) in calcium_def,
        date(Vv2) in calcium_def]) if
        :eval_rule_exp(current_date of plant, Vv2),
        season(_214445) in plant,  :(_214445\==winter),
        highly_confirmed(calcium_def) in disorder,
        material_gr11(Vv1) in operation &
r105([  method('foliage nutrition')in zinc_def,
        material_name('micro element mixture')in zinc_def,
        number(1)in zinc_def,
        date(Vv1) in zinc_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
        confirmed(zinc_def) in disorder &
r106([  method('foliage nutrition')in zinc_def,
        material_name('micro element mixture')in zinc_def,
        number(1)in zinc_def,
        date(Vv1) in zinc_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(summer) in plant,
        highly_confirmed(zinc_def) in disorder &
r107([  material_name(none)in zinc_def,
        method(advice)in zinc_def,
        number(1)in zinc_def,
        date(Vv1) in zinc_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_219100) in plant,  :(_219100\==summer),
        confirmed(zinc_def) in disorder &
r108([  material_name(none)in zinc_def,
        method(advice)in zinc_def,
        number(1)in zinc_def,
        date(Vv1) in zinc_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_220671) in plant,  :(_220671\==summer),
        highly_confirmed(zinc_def) in disorder &
r109([  material_name(none)in iron_def,
        method(advice)in iron_def,
        number(1)in iron_def,
        date(Vv1) in iron_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_222242) in plant,  :(_222242\==summer),
        confirmed(iron_def) in disorder &
r110([  material_name(none)in iron_def,
        method(advice)in iron_def,
        number(1)in iron_def,
        date(Vv1) in iron_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_223813) in plant,  :(_223813\==summer),
        highly_confirmed(iron_def) in disorder &
r111([  material_name(none)in manganese_def,
```

```
        method(advice)in manganese_def,
        number(1)in manganese_def,
        date(Vv1) in manganese_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_225384) in plant, :(_225384\==summer),
        confirmed(manganese_def) in disorder &
r112([  material_name(none)in manganese_def,
        method(advice)in manganese_def,
        number(1)in manganese_def,
        date(Vv1) in manganese_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_226955) in plant, :(_226955\==summer),
        highly_confirmed(manganese_def) in disorder &
r113([  material_name(none)in magnesium_def,
        method(advice)in magnesium_def,
        number(1)in magnesium_def,
        date(Vv1) in magnesium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_228526) in plant, :(_228526\==summer),
        confirmed(magnesium_def) in disorder &
r114([  material_name(none)in magnesium_def,
        method(advice)in magnesium_def,
        number(1)in magnesium_def,
        date(Vv1) in magnesium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(_230097) in plant, :(_230097\==summer),
        highly_confirmed(magnesium_def) in disorder &
r115([  material_name(none)in nitrogen_def,
        method(advice)in nitrogen_def,
        number(1)in nitrogen_def,
        date(Vv1) in nitrogen_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        confirmed(nitrogen_def) in disorder &
r116([  material_name(none)in nitrogen_def,
        method(advice)in nitrogen_def,
        number(1)in nitrogen_def,
        date(Vv1) in nitrogen_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        highly_confirmed(nitrogen_def) in disorder &
r117([  material_name(none)in potassium_def,
        method(advice)in potassium_def,
        number(1)in potassium_def,
        date(Vv1) in potassium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        confirmed(potassium_def) in disorder &
r118([  material_name(none)in potassium_def,
        method(advice)in potassium_def,
        number(1)in potassium_def,
        date(Vv1) in potassium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        highly_confirmed(potassium_def) in disorder &
r119([  material_name(none)in phosphorus_def,
        method(advice)in phosphorus_def,
        number(1)in phosphorus_def,
        date(Vv1) in phosphorus_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
```

```
        season(winter) in plant,
        confirmed(phosphorus_def) in disorder &
r120([  material_name(none)in phosphorus_def,
        method(advice)in phosphorus_def,
        number(1)in phosphorus_def,
        date(Vv1) in phosphorus_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        highly_confirmed(phosphorus_def) in disorder &
r121([  material_name(none)in calcium_def,
        method(advice)in calcium_def,
        number(1)in calcium_def,
        date(Vv1) in calcium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        confirmed(calcium_def) in disorder &
r122([  material_name(none)in calcium_def,
        method(advice)in calcium_def,
        number(1)in calcium_def,
        date(Vv1) in calcium_def]) if
        :eval_rule_exp(current_date of plant, Vv1),
        season(winter) in plant,
        highly_confirmed(calcium_def) in disorder &
r127([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        confirmed(mediterranean_fruit_fly) in disorder,
        current_month(4) in plant,
        material_gr12(Vv1) in operation &
r128([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        highly_confirmed(mediterranean_fruit_fly) in disorder,
        current_month(4) in plant,
        material_gr12(Vv1) in operation &
r129([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        confirmed(mediterranean_fruit_fly) in disorder,
        current_month(9) in plant,
        material_gr12(Vv1) in operation &
r130([  method('chemical spray')in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        material_name(Vv1) in mediterranean_fruit_fly,
        date(Vv2) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv2),
        highly_confirmed(mediterranean_fruit_fly) in disorder,
        current_month(9) in plant,
        material_gr12(Vv1) in operation &
r131([  material_name(none)in mediterranean_fruit_fly,
        method(advice)in mediterranean_fruit_fly,
        number(1)in mediterranean_fruit_fly,
        date(Vv1) in mediterranean_fruit_fly]) if
        :eval_rule_exp(current_date of plant, Vv1),
```

```
            confirmed(mediterranean_fruit_fly) in disorder,
            current_month(_250160) in plant,   :(_250160\==4) &
r132([      material_name(none)in mediterranean_fruit_fly,
            method(advice)in mediterranean_fruit_fly,
            number(1)in mediterranean_fruit_fly,
            date(Vv1) in mediterranean_fruit_fly]) if
            :eval_rule_exp(current_date of plant, Vv1),
            highly_confirmed(mediterranean_fruit_fly) in disorder,
            current_month(_251731) in plant,   :(_251731\==4) &
r133([      material_name(none)in mediterranean_fruit_fly,
            method(advice)in mediterranean_fruit_fly,
            number(1)in mediterranean_fruit_fly,
            date(Vv1) in mediterranean_fruit_fly]) if
            :eval_rule_exp(current_date of plant, Vv1),
            confirmed(mediterranean_fruit_fly) in disorder,
            current_month(_253302) in plant,   :(_253302\==9) &
r134([      material_name(none)in mediterranean_fruit_fly,
            method(advice)in mediterranean_fruit_fly,
            number(1)in mediterranean_fruit_fly,
            date(Vv1) in mediterranean_fruit_fly]) if
            :eval_rule_exp(current_date of plant, Vv1),
            highly_confirmed(mediterranean_fruit_fly) in disorder,
            current_month(_254873) in plant,   :(_254873\==9) &
super(rules)
}.

treat_op_determine_treat_op :: {
r1([        material_qty(200)in citrus_flower_moth,
            unit('gm/100 l water')in citrus_flower_moth]) if
            material_name('super aside') in citrus_flower_moth &
r2([        material_qty(200)in citrus_flower_moth,
            unit('gm/100 l water')in citrus_flower_moth]) if
            material_name(aikaten) in citrus_flower_moth &
r3([        material_qty(300)in citrus_flower_moth,
            unit('ml/100 l water')in citrus_flower_moth]) if
            material_name('anthio 33%') in citrus_flower_moth &
r4([        material_qty(300)in citrus_flower_moth,
            unit('ml/100 l water')in citrus_flower_moth]) if
            material_name('actellic 50%') in citrus_flower_moth &
r5([        material_qty(150)in aphids,
            unit('ml/100 l water')in aphids]) if
            material_name('malathion 57%') in aphids &
r6([        material_qty(30)in citrus_white_fly,
            unit('ml/100 l water')in citrus_white_fly]) if
            material_name('vertimec 1.8%') in citrus_white_fly &
r7([        material_qty(1.5)in aphids,
            unit('L/100 l water')in aphids]) if
            material_name('super masrona 94%') in aphids &
r8([        material_qty(1.5)in citrus_white_fly,
            unit('L/100 l water')in citrus_white_fly]) if
            material_name('super masrona 94%') in citrus_white_fly &
r9([        material_qty(1.5)in scales,
            unit('L/100 l water')in scales]) if
            material_name('super masrona 94%') in scales &
r10([       material_qty(1.5)in mealy_bug,
            unit('L/100 l water')in mealy_bug]) if
            material_name('super masrona 94%') in mealy_bug &
r11([       material_qty(1.5)in aphids,
            unit('L/100 l water')in aphids]) if
            material_name('super royal 95%') in aphids &
```

```
r12([    material_qty(1.5)in citrus_white_fly,
         unit('L/100 l water')in citrus_white_fly]) if
         material_name('super royal 95%') in citrus_white_fly &
r13([    material_qty(1.5)in scales,
         unit('L/100 l water')in scales]) if
         material_name('super royal 95%') in scales &
r14([    material_qty(1.5)in mealy_bug,
         unit('L/100 l water')in mealy_bug]) if
         material_name('super royal 95%') in mealy_bug &
r15([    material_qty(1.5)in aphids,
         unit('L/100 l water')in aphids]) if
         material_name('K.Z. 95%') in aphids &
r16([    material_qty(1.5)in citrus_white_fly,
         unit('L/100 l water')in citrus_white_fly]) if
         material_name('K.Z. 95%') in citrus_white_fly &
r17([    material_qty(1.5)in scales,
         unit('L/100 l water')in scales]) if
         material_name('K.Z. 95%') in scales &
r18([    material_qty(1.5)in mealy_bug,
         unit('L/100 l water')in mealy_bug]) if
         material_name('K.Z. 95%') in mealy_bug &
r19([    unit('L/100 l water')in aphids,
         material_qty(1.6)in aphids]) if
         material_name('Kimisol 95%') in aphids &
r20([    unit('L/100 l water')in citrus_white_fly,
         material_qty(1.6)in citrus_white_fly]) if
         material_name('Kimisol 95%') in citrus_white_fly &
r21([    unit('L/100 l water')in scales,
         material_qty(1.6)in scales]) if
         material_name('Kimisol 95%') in scales &
r22([    unit('L/100 l water')in mealy_bug,
         material_qty(1.6)in mealy_bug]) if
         material_name('Kimisol 95%') in mealy_bug &
r23a([   material_qty(25)in leafminer_op1,
         unit('ml + 25 ml/100 l water')in leafminer_op1]) if
         material_name('vertimec + super masrona 94%') in leafminer_op1 &
r23b([   material_qty(25)in leafminer_op2,
         unit('ml + 25 ml/100 l water')in leafminer_op2]) if
         material_name('vertimec + super masrona 94%') in leafminer_op2 &
r23c([   material_qty(25)in leafminer_op3,
         unit('ml + 25 ml/100 l water')in leafminer_op3]) if
         material_name('vertimec + super masrona 94%') in leafminer_op3 &
r24a([   material_qty(25)in leafminer_op1,
         unit('ml + 25 ml/100 l water')in leafminer_op1]) if
         material_name('vertimec + super royal oil 95%') in leafminer_op1 &
r24b([   material_qty(25)in leafminer_op2,
         unit('ml + 25 ml/100 l water')in leafminer_op2]) if
         material_name('vertimec + super royal oil 95%') in leafminer_op2 &
r24c([   material_qty(25)in leafminer_op3,
         unit('ml + 25 ml/100 l water')in leafminer_op3]) if
         material_name('vertimec + super royal oil 95%') in leafminer_op3 &
r25a([   material_qty(25)in leafminer_op1,
         unit('ml + 25 ml/100 l water')in leafminer_op1]) if
         material_name('vertimec + K.Z oil 95%') in leafminer_op1 &
r25b([   material_qty(25)in leafminer_op2,
         unit('ml + 25 ml/100 l water')in leafminer_op2]) if
         material_name('vertimec + K.Z oil 95%') in leafminer_op2 &
r25c([   material_qty(25)in leafminer_op3,
         unit('ml + 25 ml/100 l water')in leafminer_op3]) if
         material_name('vertimec + K.Z oil 95%') in leafminer_op3 &
```

```
r26a([   material_qty(25)in leafminer_op1,
         unit('ml + 25 ml/100 l water')in leafminer_op1]) if
         material_name('vertimec + Kimisol oil 95%') in leafminer_op1 &
r26b([   material_qty(25)in leafminer_op2,
         unit('ml + 25 ml/100 l water')in leafminer_op2]) if
         material_name('vertimec + Kimisol oil 95%') in leafminer_op2 &
r26c([   material_qty(25)in leafminer_op3,
         unit('ml + 25 ml/100 l water')in leafminer_op3]) if
         material_name('vertimec + Kimisol oil 95%') in leafminer_op3 &
r27([    material_qty(10)in gummosis,
         unit('gm/1 l water')in gummosis]) if
         material_name(potassium_permenganat) in gummosis &
r28([    material_qty(1)in gummosis,
         unit('kg CuSo4 + 2 Kg CaO + 10 L water')in gummosis]) if
         material_name('bordeaux past') in gummosis &
r29a([   material_qty(20)in wilt_root_rot_op1,
         unit('gm/tree')in wilt_root_rot_op1]) if
         material_name(topsin) in wilt_root_rot_op1 &
r29b([   material_qty(20)in wilt_root_rot_op2,
         unit('gm/tree')in wilt_root_rot_op2]) if
         material_name(topsin) in wilt_root_rot_op2 &
r30a([   material_qty(150)in ganoderma_rot_op1,
         unit('gm/100 l water')in ganoderma_rot_op1]) if
         material_name(topsin) in ganoderma_rot_op1 &
r30b([   material_qty(150)in ganoderma_rot_op2,
         unit('gm/100 l water')in ganoderma_rot_op2]) if
         material_name(topsin) in ganoderma_rot_op1 &
r31([    material_qty(500)in lichens,
         unit('gm/100 l water')in lichens]) if
         material_name(copper_oxychloride) in lichens &
r32([    material_qty(250)in lichens,
         unit('gm/100 l water')in lichens]) if
         material_name('cuprus K.Z 50%') in lichens &
r33([    material_qty(500)in lichens,
         unit('gm/100 l water')in lichens]) if
         material_name('pory coper 50%') in lichens &
r34([    material_qty(500)in lichens,
         unit('gm/100 l water')in lichens]) if
         material_name('pro coper 50%') in lichens &
r35([    material_qty(500)in lichens,
         unit('gm/100 l water')in lichens]) if
         material_name('copox 50%') in lichens &
r36([    material_qty(1)in lichens,
         unit('Kg Cu So4 + 1.5 CaO/100 l water')in lichens]) if
         material_name('caprimex 98%') in lichens &
r37([    material_qty(350)in lichens,
         unit('gm/100 l water')in lichens]) if
         material_name('halomac 65%') in lichens &
r38a([   material_qty(50)in flat_mite_op1,
         unit('ml + 150 ml/100 l water')in flat_mite_op1]) if
         material_name('ortis 5% sc + kz oil') in flat_mite_op1 &
r38b([   material_qty(50)in flat_mite_op2,
         unit('ml + 150 ml/100 l water')in flat_mite_op2]) if
         material_name('ortis 5% sc + kz oil') in flat_mite_op2 &
r39a([   material_qty(50)in brown_mite_op1,
         unit('ml + 150 ml/100 l water')in brown_mite_op1]) if
         material_name('ortis 5% sc + kz oil') in brown_mite_op1 &
r39b([   material_qty(50)in brown_mite_op2,
         unit('ml + 150 ml/100 l water')in brown_mite_op2]) if
         material_name('ortis 5% sc + kz oil') in brown_mite_op2 &
```

```
r40a([   material_qty(100)in rust_mite_op1,
         unit('ml + 150 ml/100 l water')in rust_mite_op1]) if
         material_name('ortis 5% sc + kz oil') in rust_mite_op1 &
r40b([   material_qty(100)in rust_mite_op2,
         unit('ml + 150 ml/100 l water')in rust_mite_op2]) if
         material_name('ortis 5% sc + kz oil') in rust_mite_op2 &
r41a([   material_qty(100)in bud_mite_op1,
         unit('ml + 150 ml/100 l water')in bud_mite_op1]) if
         material_name('ortis 5% sc + kz oil') in bud_mite_op1 &
r41b([   material_qty(100)in bud_mite_op2,
         unit('ml + 150 ml/100 l water')in bud_mite_op2]) if
         material_name('ortis 5% sc + kz oil') in bud_mite_op2 &
r42a([   material_qty(40)in rust_mite_op1,
         unit('ml/100 l water')in rust_mite_op1]) if
         material_name('neron 50%') in rust_mite_op1 &
r42b([   material_qty(40)in rust_mite_op2,
         unit('ml/100 l water')in rust_mite_op2]) if
         material_name('neron 50%') in rust_mite_op2 &
r43a([   material_qty(40)in bud_mite_op1,
         unit('ml/100 l water')in bud_mite_op1]) if
         material_name('neron 50%') in bud_mite_op1 &
r43b([   material_qty(40)in bud_mite_op2,
         unit('ml/100 l water')in bud_mite_op2]) if
         material_name('neron 50%') in bud_mite_op2 &
r44a([   material_qty(30)in rust_mite_op1,
         unit('ml + 250 ml/100 L water')in rust_mite_op1]) if
         material_name('vertimec 1.8% + kz oil') in rust_mite_op1 &
r44b([   material_qty(30)in rust_mite_op2,
         unit('ml + 250 ml/100 L water')in rust_mite_op2]) if
         material_name('vertimec 1.8% + kz oil') in rust_mite_op2 &
r45a([   material_qty(30)in bud_mite_op1,
         unit('ml + 250 ml/100 L water')in bud_mite_op1]) if
         material_name('vertimec 1.8% + kz oil') in bud_mite_op1 &
r45b([   material_qty(30)in bud_mite_op2,
         unit('ml + 250 ml/100 L water')in bud_mite_op2]) if
         material_name('vertimec 1.8% + kz oil') in bud_mite_op2 &
r46a([   material_qty(30)in flat_mite_op1,
         unit('ml + 250 ml/100 L water')in flat_mite_op1]) if
         material_name('vertimec 1.8% + kz oil') in flat_mite_op1 &
r46b([   material_qty(30)in flat_mite_op2,
         unit('ml + 250 ml/100 L water')in flat_mite_op2]) if
         material_name('vertimec 1.8% + kz oil') in flat_mite_op2 &
r47a([   material_qty(30)in brown_mite_op1,
         unit('ml + 250 ml/100 L water')in brown_mite_op1]) if
         material_name('vertimec 1.8% + kz oil') in brown_mite_op1 &
r47b([   material_qty(30)in brown_mite_op2,
         unit('ml + 250 ml/100 L water')in brown_mite_op2]) if
         material_name('vertimec 1.8% + kz oil') in brown_mite_op2 &
r48a([   material_qty(100)in flat_mite_op1,
         unit('ml/100 l water')in flat_mite_op1]) if
         material_name(pride) in flat_mite_op1 &
r48b([   material_qty(100)in flat_mite_op2,
         unit('ml/100 l water')in flat_mite_op2]) if
         material_name(pride) in flat_mite_op2 &
r49a([   material_qty(100)in brown_mite_op1,
         unit('ml/100 l water')in brown_mite_op1]) if
         material_name(pride) in brown_mite_op1 &
r49b([   material_qty(100)in brown_mite_op2,
         unit('ml/100 l water')in brown_mite_op2]) if
         material_name(pride) in brown_mite_op2 &
```

```
r50a([    material_qty(17)in citrus_nematude_op1,
          unit('kg/feddan')in citrus_nematude_op1]) if
          material_name('temic 15%') in citrus_nematude_op1 &
r50b([    material_qty(17)in citrus_nematude_op2,
          unit('kg/feddan')in citrus_nematude_op2]) if
          material_name('temic 15%') in citrus_nematude_op2 &
r51a([    material_qty(40)in citrus_nematude_op1,
          unit('kg/feddan')in citrus_nematude_op1]) if
          material_name('furidan 10%') in citrus_nematude_op1 &
r51b([    material_qty(40)in citrus_nematude_op2,
          unit('kg/feddan')in citrus_nematude_op2]) if
          material_name('furidan 10%') in citrus_nematude_op2 &
r52a([    material_qty(24)in citrus_nematude_op1,
          unit('kg/feddan')in citrus_nematude_op1]) if
          material_name('ragbi 10%') in citrus_nematude_op1 &
r52b([    material_qty(24)in citrus_nematude_op2,
          unit('kg/feddan')in citrus_nematude_op2]) if
          material_name('ragbi 10%') in citrus_nematude_op2 &
r53a([    material_qty(4)in citrus_nematude_op1,
          unit('L/feddan')in citrus_nematude_op1]) if
          material_name(vaydete) in citrus_nematude_op1 &
r53b([    material_qty(4)in citrus_nematude_op2,
          unit('L/feddan')in citrus_nematude_op2]) if
          material_name(vaydete) in citrus_nematude_op2 &
r54([     material_qty(2.5)in mealy_bug,
          unit('L/100 l water')in mealy_bug]) if
          material_name('bolum oil 80%') in mealy_bug &
r55([     material_qty(2.5)in mealy_bug,
          unit('L/100 l water')in mealy_bug]) if
          material_name('royal oil 80%') in mealy_bug &
r56([     material_qty(2.5)in mealy_bug,
          unit('L/100 l water')in mealy_bug]) if
          material_name('misrona oil 80%') in mealy_bug &
r57([     material_qty(2.5)in mealy_bug,
          unit('L/100 l water')in mealy_bug]) if
          material_name('agro oil 80%') in mealy_bug &
r58([     material_qty(2.0)in mealy_bug,
          unit('L/100 l water')in mealy_bug]) if
          material_name('focal oil 82%') in mealy_bug &
r59([     material_qty(2.5)in scales,
          unit('L/100 l water')in scales]) if
          material_name('bolum oil 80%') in scales &
r60([     material_qty(2.5)in scales,
          unit('L/100 l water')in scales]) if
          material_name('royal oil 80%') in scales &
r61([     material_qty(2.5)in scales,
          unit('L/100 l water')in scales]) if
          material_name('misrona oil 80%') in scales &
r62([     material_qty(2.5)in scales,
          unit('L/100 l water')in scales]) if
          material_name('agro oil 80%') in scales &
r63([     material_qty(2.0)in scales,
          unit('L/100 l water')in scales]) if
          material_name('focal oil 82%') in scales &
r64([     material_qty(100)in mediterranean_fruit_fly,
          unit('ml + 500 ml/100 l water')in mediterranean_fruit_fly]) if
          material_name('malthion 57% + policure') in mediterranean_fruit_fly &
r65([     material_qty(500)in mediterranean_fruit_fly,
          unit('ml + L/100 l water')in mediterranean_fruit_fly]) if
          material_name('libacid 50% + bominal') in mediterranean_fruit_fly &
```

```
r66([    material_qty(2)in calcium_def,
         unit('kg/100 l water')in calcium_def]) if
         material_name('calcium nitrate') in calcium_def &
r67([    material_qty(0.5)in calcium_def,
         unit('kg/100 l water')in calcium_def]) if
         material_name('calcium chloride') in calcium_def &
r68([    material_qty(1.5)in potassium_def,
         unit('kg/100 l water')in potassium_def]) if
         material_name(potassium_nitrate) in potassium_def &
r69([    material_qty(2)in potassium_def,
         unit('kg/100 l water')in potassium_def]) if
         material_name(potassium_sulfate) in potassium_def &
r70([    material_qty(1)in phosphorus_def,
         unit('kg/100 l water')in phosphorus_def]) if
         material_name('triple phosphate') in phosphorus_def &
r71([    material_qty(1)in nitrogen_def,
         unit('kg/100 l water')in nitrogen_def]) if
         material_name(urea) in nitrogen_def &
r72([    material_qty(1.5)in nitrogen_def,
         unit('kg/100 l water')in nitrogen_def]) if
         material_name('ammonium nitrate') in nitrogen_def &
r73([    material_qty(0.5)in magnesium_def,
         unit('kg/100 l water')in magnesium_def]) if
         material_name(magnesium_sulfate) in magnesium_def &
r74([    material_qty(0)in treat_op,
         unit('as below')in treat_op]) if
         material_name('micro element mixture') in treat_op &
super(rules)
}.
enhanced_by :: {
r1([     advice('Avoid excess of nitrogen fertilizers and organic manure near the trunk. Also, avoid
excess irrigation water near the trunk.')in gummosis]) if
         method(painting) in gummosis,
         season(winter) in plant &
r2a([    advice('Application of acaricides is recommended at 20 % infestation, in general. Spot
spraying localized infestation is good practice and tractor drawn equipment with agitator is often the
ideal machine for application. Spraying should be as a mist, tacking umbrella shape at lower pressure
and as possible over the entire tree')in rust_mite_op1]) if
         method('chemical spray') in rust_mite_op1,
         season(summer) in plant &
r2b([    advice('Application of acaricides is recommended at 20 % infestation, in general. Spot
spraying localized infestation is good practice and tractor drawn equipment with agitator is often the
ideal machine for application. Spraying should be as a mist, tacking umbrella shape at lower pressure
and as possible over the entire tree')in rust_mite_op2]) if
         method('chemical spray') in rust_mite_op2,
         season(summer) in plant &
r3a([    advice('The treatment at this time is not recommended. Time of chemical control in late of
April, in case of infestation')in rust_mite_op1]) if
         method('chemical spray') in rust_mite_op1,
         season(_60191) in plant,   :(_60191\==summer) &
r3b([    advice('The treatment at this time is not recommended. Time of chemical control in late of
April, in case of infestation')in rust_mite_op2]) if
         method('chemical spray') in rust_mite_op2,
         season(_60191) in plant,   :(_60191\==summer) &
r4a1([   advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in bud_mite_op1,
         advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible')in bud_mite_op1]) if
         method('chemical spray') in bud_mite_op1,
         current_week(_61304) in plant,     :(_61304>=7),
```

```
        current_week(_61567) in plant,      :(_61567=<22) &
r4a2([  advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in bud_mite_op2,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_61304) in plant,      :(_61304>=7),
        current_week(_61567) in plant,      :(_61567=<22) &
r4b1([  advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in bud_mite_op1,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible')in bud_mite_op1]) if
        method('chemical spray') in bud_mite_op1,
        current_week(_62680) in plant,      :(_62680>=35),
        current_week(_62943) in plant,      :(_62943=<44) &
r4b2([  advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in bud_mite_op2,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_62680) in plant,      :(_62680>=35),
        current_week(_62943) in plant,      :(_62943=<44) &
r5a1([  advice('The treatment at this time is not recommended. Time of chemical control in late of
February, in case of infestation')in bud_mite_op1]) if
        method('chemical spray') in bud_mite_op1,
        current_week(_63926) in plant,      :(_63926>0),
        current_week(_64189) in plant,      :(_64189<7) &
r5a2([  advice('The treatment at this time is not recommended. Time of chemical control in late of
February, in case of infestation')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_63926) in plant,      :(_63926>0),
        current_week(_64189) in plant,      :(_64189<7) &
r5b1([  advice('The treatment at this time is not recommended. Time of chemical control in late of
February, in case of infestation')in bud_mite_op1]) if
        method('chemical spray') in bud_mite_op1,
        current_week(_65172) in plant,      :(_65172>22),
        current_week(_65435) in plant,      :(_65435<35) &
r5b2([  advice('The treatment at this time is not recommended. Time of chemical control in late of
February, in case of infestation')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_65172) in plant,      :(_65172>22),
        current_week(_65435) in plant,      :(_65435<35) &
r5c1([  advice('The treatment at this time is not recommended. Time of chemical control in late of
February, in case of infestation')in bud_mite_op1]) if
        method('chemical spray') in bud_mite_op1,
        current_week(_66418) in plant,      :(_66418>44),
        current_week(_66681) in plant,      :(_66681<54) &
r5c2([  advice('The treatment at this time is not recommended. Time of chemical control in late of
February, in case of infestation')in bud_mite_op2]) if
        method('chemical spray') in bud_mite_op2,
        current_week(_66418) in plant,      :(_66418>44),
        current_week(_66681) in plant,      :(_66681<54) &
r6a([   advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in brown_mite_op1,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible from upwards to downwards')in brown_mite_op1]) if
        method('chemical spray') in brown_mite_op1,
        season(summer) in plant &
r6b([   advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in brown_mite_op2,
```

```
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible from upwards to downwards')in brown_mite_op2]) if
        method('chemical spray') in brown_mite_op2,
        season(summer) in plant &
r7a([   advice('The treatment at this time is not recommended. Time of chemical control in late of
May, in case of infestation')in brown_mite_op1]) if
        method('chemical spray') in brown_mite_op1,
        season(_68618) in plant,   :(_68618\==summer) &
r7b([   advice('The treatment at this time is not recommended. Time of chemical control in late of
May, in case of infestation')in brown_mite_op2]) if
        method('chemical spray') in brown_mite_op2,
        season(_68618) in plant,   :(_68618\==summer) &
r8a([   advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in flat_mite_op1,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible from downwards to up words and pointing to the core of tree')in flat_mite_op1]) if
        method('chemical spray') in flat_mite_op1,
        season(summer) in plant &
r8b([   advice('Spot spraying localized infestation is good practice and tractor drawn equipment with
agitator is often the ideal machine for application')in flat_mite_op2,
        advice('Spraying should be as a mist, tacking umbrella shape at lower pressure and as far as
possible from downwards to up words and pointing to the core of tree')in flat_mite_op2]) if
        method('chemical spray') in flat_mite_op2,
        season(summer) in plant &
r9a([   advice('The treatment at this time is not recommended. Time of chemical control in late of
April, in case of infestation')in flat_mite_op1]) if
        method('chemical spray') in flat_mite_op1,
        season(_70555) in plant,   :(_70555\==summer) &
r9b([   advice('The treatment at this time is not recommended. Time of chemical control in late of
April, in case of infestation')in flat_mite_op2]) if
        method('chemical spray') in flat_mite_op2,
        season(_70555) in plant,   :(_70555\==summer) &
r10([   advice('It is important to check the soil salinity, and in case of high salinity the leaching is
recommended')in magnesium_def]) if
        method('foliage nutrition') in magnesium_def,
        season(summer) in plant &
r11([   advice('The micro elements mixture is formulated, for every 100 lt water, as follow :  30 gm
Iron Chelate (EDTA) + 30 gm Zinc Chelate + 75 Mang. Chelate  + 6 gm Copper Sulfate + 30
Magnesium Sulfate + 0.3 gm Borax')in manganese_def]) if
        method('foliage nutrition') in manganese_def,
        season(summer) in plant &
r12([   advice('The micro elements mixture is formulated, for every 100 lt water , as follow :  150 gm
Iron Chelate (EDTA) + 30 gm Zinc Chelate + 15 Mang. Chelate  + 6 gm Copper Sulfate + 30
Magnesium Sulfate + 0.3 gm Borax')in iron_def]) if
        method('foliage nutrition') in iron_def,
        season(summer) in plant &
r13([   advice('The micro elements mixture is formulated, for every 100 lt water, as follow:  30 gm
Iron Chelate (EDTA) + 150 gm Zinc Chelate + 15 Mang. Chelate  + 6 gm Copper Sulfate + 30
Magnesium Sulfate + 0.3 gm Borax')in zinc_def]) if
        method('foliage nutrition') in zinc_def,
        season(summer) in plant &
r14([   advice('No foliage application during the flowering stage and fruit setting')in zinc_def]) if
        method(advice) in zinc_def,
        season(spring) in plant &
r15([   advice('No foliage application during the flowering stage and fruit setting')in iron_def]) if
        method(advice) in iron_def,
        season(spring) in plant &
r16([   advice('No foliage application during the flowering stage and fruit setting')in manganese_def])
if
        method(advice) in manganese_def,
```

```
        season(spring) in plant &
r17([   advice('No foliage application during the flowering stage and fruit setting')in
magnesium_def]) if
        method(advice) in magnesium_def,
        season(spring) in plant &
r18([   advice('No foliage application during the fruits collecting period')in zinc_def]) if
        method(advice) in zinc_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r19([   advice('No foliage application during the fruits collecting period')in iron_def]) if
        method(advice) in iron_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r20([   advice('No foliage application during the fruits collecting period')in manganese_def]) if
        method(advice) in manganese_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r21([   advice('No foliage application during the fruits collecting period')in magnesium_def]) if
        method(advice) in magnesium_def,
        (       season(autumn) in plant
        ;       season(winter) in plant
        ), ! &
r22([   advice('Substitute the nitrogen quantity in the fertilization expert system recommendation by
its equivalence of calcium nitrate')in calcium_def]) if
        method('foliage nutrition') in calcium_def,
        season(_82672) in plant,    :(_82672\==winter) &
r23([   advice('No significant response of trees to foliar application during winter. Therefore treat
your trees in the beginning of spring')in nitrogen_def]) if
        method(advice) in nitrogen_def,
        season(winter) in plant &
r24([   advice('No significant response of trees to foliar application during winter. Therefore treat
your trees in the beginning of spring')in potassium_def]) if
        method(advice) in potassium_def,
        season(winter) in plant &
r25([   advice('No significant response of trees to foliar application during winter. Therefore treat
your trees in the beginning of spring')in phosphorus_def]) if
        method(advice) in phosphorus_def,
        season(winter) in plant &
r26([   advice('No significant response of trees to foliar application during winter. Therefore treat
your trees in the beginning of spring')in calcium_def]) if
        method(advice) in calcium_def,
        season(winter) in plant &
r27([   advice('Spraying two branches only in each tree and collects infested fruits and bury it.')in
mediterranean_fruit_fly]) if
        method('chemical spray') in mediterranean_fruit_fly,
        current_month(4) in plant &
r28([   advice('Spraying two branches only in each tree and collects infested fruits and bury it.')in
mediterranean_fruit_fly]) if
        method('chemical spray') in mediterranean_fruit_fly,
        current_month(9) in plant &
r29a([  advice('Collect infested fruits and bury it.')in mediterranean_fruit_fly,
        advice('The treatment at this time is not recommended.')in mediterranean_fruit_fly]) if
        method(advice) in mediterranean_fruit_fly,
        current_month(_88859) in plant,     :(_88859\==4) &
r29b([  advice('Collect infested fruits and bury it.')in mediterranean_fruit_fly,
        advice('The treatment at this time is not recommended.')in mediterranean_fruit_fly]) if
        method(advice) in mediterranean_fruit_fly,
```

```
        current_month(_89946) in plant,     :(_89946\==9) &
r30([   advice('Increase quantity of fertilizer application by 50% and incrementally increased up to
100% or until disappearance of nutrient deficiency observations, then apply the recommendations
given by the fertilization expert system')in nitrogen_def]) if
        method('foliage nutrition') in nitrogen_def,
        nitrogen_infestation('very low') in disorder,
        season(_91085) in plant,    :(_91085\==winter) &
r31([   advice('Increase quantity of fertilizer application by 25% and incrementally increased up to
50% or until disappearance of nutrient deficiency observations, then apply the recommendations given
by the fertilization expert system')in nitrogen_def]) if
        method('foliage nutrition') in nitrogen_def,
        nitrogen_infestation(low) in disorder,
        season(_92224) in plant,    :(_92224\==winter) &
r32([   advice('Increase quantity of fertilizer application by 50% and incrementally increased up to
100% or until disappearance of nutrient deficiency observations, then apply the recommendations
given by the fertilization expert system')in potassium_def]) if
        method('foliage nutrition') in potassium_def,
        nitrogen_infestation('very low') in disorder,
        season(_93363) in plant,    :(_93363\==winter) &
r33([   advice('Increase quantity of fertilizer application by 25% and incrementally increased up to
50% or until disappearance of nutrient deficiency observations, then apply the recommendations given
by the fertilization expert system')in potassium_def]) if
        method('foliage nutrition') in potassium_def,
        nitrogen_infestation(low) in disorder,
        season(_94502) in plant,    :(_94502\==winter) &
r34([   advice('Increase quantity of fertilizer application by 50% and incrementally increased up to
100% or until disappearance of nutrient deficiency observations, then apply the recommendations
given by the fertilization expert system')in phosphorus_def]) if
        method('foliage nutrition') in phosphorus_def,
        nitrogen_infestation('very low') in disorder,
        season(_94502) in plant,    :(_94502\==winter) &
r35([   advice('Increase quantity of fertilizer application by 25% and incrementally increased up to
50% or until disappearance of nutrient deficiency observations, then apply the recommendations given
by the fertilization expert system')in phosphorus_def]) if
        method('foliage nutrition') in phosphorus_def,
        nitrogen_infestation(low) in disorder,
        season(_96673) in plant,    :(_96673\==winter) &
r36([   advice('Good caring the diseased trees; i.e. better agriculture practices and fertilization to
extend the productive life of tree when yield becomes not economic, the diseased trees must be
replaced. Use certified transplants')in psorosis]) if
        method(advice) in psorosis &
r37([   advice('Infected young trees should be replaced by other healthy plants. Use certified
transplants')in stubborn]) if
        method(advice) in stubborn &
r38([   advice('Infected young trees should be replaced by other healthy plants. Use certified
transplants')in impieetratura]) if
        method(advice) in impieetratura &
r39([   advice('Improve the agriculture practices')in anthracnose]) if
        method(advice) in anthracnose &
r40([   advice('Improve the agriculture practices')in alternaria_leaves_spot]) if
        method(advice) in alternaria_leaves_spot &
r41([   advice('Control the insects that produce the honey dew')in sooty_mold]) if
        method(advice) in sooty_mold &
r42([   advice('Collect infected fruits and bury it. Perform the suitable agriculture practices')in
alternaria_rot]) if
        method(advice) in alternaria_rot &
r43([   advice('The diseased trees must be replaced')in armillaria_root_rot]) if
        method(advice) in armillaria_root_rot &
r44([   advice('No treatment for this phenomena where its economic importance is limited')in
gum_spots]) if
```

```
        method(advice) in gum_spots &
r45([   advice('Improve the growth of trees to protect the fruits from direct sun light')in sun_burn]) if
        method(advice) in sun_burn &
r46([   advice('Manage the irrigation and increase the fertilization quantity of Potassium')in
fruit_cracking]) if
        method(advice) in fruit_cracking &
r47([   advice('Manage the irrigation and increase the fertilization quantity of Potassium')in
fruit_creasing]) if
        method(advice) in fruit_creasing &
r48([   advice('Cultivate plant tarps for scarab like faba-beans, turnip and cauliflower')in rose_scarab,
        advice('Picking up the insects twice a day')in rose_scarab,
        advice('Spread watercolor traps at the rate 35 to 40 traps per feddan')in rose_scarab,
        advice('Use compost organic manure')in rose_scarab]) if
        method(advice) in rose_scarab &
r49([   advice('Use irrigation program to add leaching requirements')in salt_injury]) if
        method(advice) in salt_injury &
r50([   advice('No treatment for this pest, such that it is not important economically')in
green_stink_bug]) if
        method(advice) in green_stink_bug &
r51([   advice('The gum pocked must be removed with sharp knife, the wound and exposed tissues
must be disinfected with solution')in gummosis]) if
        method(disinfection) in gummosis &
r52a([  advice('Remove fungal growths and painting the wound by Bordeaux past then spray the
green area of trees.  The formula of Bordeaux past is:  1 kg cuso + 2 kg cad  + water')in
ganoderma_rot_op1]) if
        method('chemical spray') in ganoderma_rot_op1 &
r52b([  advice('Remove fungal growths and painting the wound by Bordeaux past then spray the
green area of trees.  The formula of Bordeaux past is:  1 kg cuso + 2 kg cad  + water')in
ganoderma_rot_op2]) if
        method('chemical spray') in ganoderma_rot_op2 &
r53([   advice('Spray the infested trees only')in aphids,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch without
direct application')in aphids]) if
        method('chemical spray') in aphids,
        :( \+ method('chemical spray') in citrus_white_fly) &
r54([   advice('Spray the infested trees only')in citrus_white_fly,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch without
direct application')in citrus_white_fly]) if
        :( \+ method('chemical spray') in aphids),
        method('chemical spray') in citrus_white_fly &
r55([   advice('Spray the infested trees only')in aphids,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch without
direct application')in aphids,
        advice('This operation used as shared treatment for aphids and citrus white fly')in aphids]) if
        method('chemical spray') in aphids,
        method('chemical spray') in citrus_white_fly &
r56([   advice('Spray the infested trees only')in citrus_white_fly,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch without
direct application')in citrus_white_fly,
        advice('This operation used as shared treatment for aphids and citrus white fly')in
citrus_white_fly]) if
        method('chemical spray') in aphids,
        method('chemical spray') in citrus_white_fly &
r57([   advice('Spray trees of entire farm')in citrus_flower_moth,
        advice('The pressure of spraying motor must not exceed 100 pound per square inch without
direct application')in citrus_flower_moth]) if
        method('chemical spray') in citrus_flower_moth &
r58([   advice('Lichens control includes good agricultural practices; i.e. pruning and avoid excess
irrigation water')in lichens]) if
        method('chemical spray') in lichens &
```

```
r59([     advice('Use fit spraying motor with good mixing. The trees must be completely washed.')in
scales]) if
        method('chemical spray') in scales &
r60([     advice('Use fit spraying motor with good mixing. The trees must be completely washed.')in
mealy_bug]) if
        method('chemical spray') in mealy_bug &
r61a([    advice('You must follow this operation by light irrigation to avoid application of fruit bearing
trees')in citrus_nematude_op1]) if
        method('soil treatment') in citrus_nematude_op1 &
r61b([    advice('You must follow this operation by light irrigation to avoid application of fruit bearing
trees')in citrus_nematude_op2]) if
        method('soil treatment') in citrus_nematude_op2 &
super(rules)
}.
```

5.2 Inferance layer

```
/* File name : treat_inference.pl*/
    :- ensure_loaded('$KROL/lib/krol_init').
    treat_inference :: {

        instantiate :-
        treated_by :: conclude_all &

        assign :-
        treat_op_determine_treat_op :: conclude_all,
        enhanced_by :: conclude_all &


        order :-
                :orderM&
    super(krol_init)
    }.


/* File name : order.pl*/

    date_sort :: {

        '<'(I, J) :-
                I = [_,_,_,_,X,_,_,_,_],
                J = [_,_,_,_,Y,_,_,_,_],
                X = [], Y = [_,_,_], ! &
        '<'([_,_,_,_,X,_,_,_,_], [_,_,_,_,Y,_,_,_,_]) :-
                :atom(X), Y = [_,_,_], ! &
        '<'([_,_,_,_,X,_,_,_,_], [_,_,_,_,Y,_,_,_,_]) :-
                :compare_date(<, X, Y)
    }.
    treated_before :: {

        '<'(X, Y) :-
                insects :: descendant(X),
                (       nematode :: descendant(Y)
                ;       nutrition_def :: descendant(Y)
                ;       Y = lichens
                ), !
    }.

    sort(Type) :: {
         :- :use_module(library(lists), [append/3]) &
```

```
        qsort([], []) &
        qsort([P|L], S) :-
                partition(L, P, Small, Large),
                qsort(Small, S0),
                qsort(Large, S1),
                :append(S0, [P|S1], S) &

        partition([], _P, [], []) &
        partition([X|L1], P, Small, Large) :-
                (   Type :: '<'(X, P) ->
                   Small = [X|Small1], Large = Large1
                ;   Small = Small1, Large = [X|Large1]
                ),
                partition(L1, P, Small1, Large1)
}.



orderM :-
        findall(O,
                        (               treat_op :: leaf(O),
                                        O :: get(number(X)),
                                        X \== []
                        ), List),
        List = [_,_|_], !,
        sort(treated_before) :: qsort(List, List1),
        ordernumber(List1,1),
        satisfy_3days(List1).

orderM.



ordernumber([],_).
ordernumber([O|ListT],N):-
        O :: set(number(N)),
        N1 is N + 1,
        ordernumber(ListT,N1).

satisfy_3days([]).
satisfy_3days([_]) :- !.

satisfy_3days([gummosis_op1, gummosis_op2|List]) :- !,
        gummosis_op1 :: get(date(Date1)),
        gummosis_op2 :: get(date(Date1)),
        satisfy_3days([gummosis_op2|List]).


satisfy_3days([O1, O2|List]) :-

        O1 :: get(material_name([none|_]))->
        satisfy_3days([O2|List]);
(
        O1 :: get(date(Date1)),
        (               Date1 = [] ->
                        satisfy_3days([O2|List])
        ;               O2 :: get(date(Date2)),
                        (               Date2 = [] ->
                                        satisfy_3days([O1|List])
```

```
                ;       diference(Date2, Date1, _, Days),
                        (       Date1 = Date2 ->
                                new_date(Date2,[3,0,0], Date2x),
                                O2 ::  set_value(date(Date2x)),
                                add_three_days(List, 3)
                        ;       compare_date(<, Date2, Date1) ->
                                Ds is Days + 3,
                                new_date(Date2,[Ds,0,0], Date2x),
                                O2 ::  set_value(date(Date2x)),
                                add_three_days(List, Ds)
                        ;       Days < 3 ->
                                Ds is 3 - Days,
                                new_date(Date2,[3,0,0], Date2x),
                                O2 ::  set_value(date(Date2x)),
                                add_three_days(List, 3)
                        ;       true
                        ),
                        satisfy_3days([O2|List])
                )
        )
    ).

    add_three_days([], _).
    add_three_days([O|Os], Ds) :-
        O :: get(date(Date2)),
        (       Date2 = [] ->
                true
        ;       new_date(Date2,[Ds,0,0], Date2x),
                O ::  set_value(date(Date2x))
        ),
        add_three_days(Os, Ds).
```

5.3 Task layer

Treatment subsystem

```
/* File name : treat_task.pl*/
treat_task :: {
super(krol_init)
}.
treat_task_transfer :: {
super(treat_task)
}.
treat_task_uncondional :: {
start_inference :-
                treat_inference :: instantiate,
                krol_init :: set(mode(un)),
                treat_inference :: assign,
                treat_inference :: order,
                disorder :: get(confirmed(L1)),
                disorder :: get(highly_confirmed(L2)),
                :append(L1,L2,Dis),
                :get_treat(Dis)&
super(treat_task)
}.

treat_task_condional :: {
super(treat_task)
}.
treat_task_repetitive :: {
super(treat_task)
```

```
}.
treat_task_user :: {
super(treat_task)
}.
```

6. DataBase

```
/* File name : citex4.pl */
:- ensure_loaded('$KROL/lib/oodbc').
citex4ds :: {
        server(citex4ds) &
        uid('') &
        pwd('') &
        super(oodbc)
}.
soil_ref_table :: {
        tab(soil_ref_table) &
        col(1, gid, 'SHORT', 2) &
        col(2, did, 'SHORT', 2) &
        col(3, texture, 'TEXT', 50) &
        col(4, water_table_level, 'SINGLE', 4) &
        col(5, ec, 'SINGLE', 4) &
        col(6, ph, 'SINGLE', 4) &
        col(8, fc, 'SINGLE', 4) &
        col(9, pwp, 'SINGLE', 4) &
        condition_item(citex4ds, soil_ref_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        condition_item(citex4ds, soil_ref_table, select, did, =, 'did of farm_data ', 'SHORT') &
        query_fs(citex4ds, soil_ref_table, ['All Fields']) &
        sql_select(soil_ref_table, ['SELECT', '*', 'FROM soil_ref_table WHERE','gid = ',
_66993,and,'did = ', _67490]) :-
                farm_data :: get_value(gid(_66993)),
                farm_data :: get_value(did(_67490)) &
        super(citex4ds)
}.
water_ref_table :: {
        tab(water_ref_table) &
        col(1, gid, 'SHORT', 2) &
        col(2, did, 'SHORT', 2) &
        col(3, eciw, 'SINGLE', 4) &
        condition_item(citex4ds, water_ref_table, select, did, =, 'did of farm_data ', 'SHORT') &
        condition_item(citex4ds, water_ref_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        query_fs(citex4ds, water_ref_table, ['All Fields']) &
        sql_select(water_ref_table, ['SELECT', '*', 'FROM water_ref_table WHERE','did = ',
_71188,and,'gid = ', _71685]) :-
                farm_data :: get_value(did(_71188)),
                farm_data :: get_value(gid(_71685)) &
        super(citex4ds)
}.
climate_ref_table :: {
        tab(climate_ref_table) &
        col(1, gid, 'SHORT', 2) &
        col(2, did, 'SHORT', 2) &
        col(3, month, 'TEXT', 50) &
        col(4, avg_tc, 'SINGLE', 4) &
        col(5, avg_rh, 'SINGLE', 4) &
        col(6, ash, 'SINGLE', 4) &
        col(7, msh, 'SINGLE', 4) &
        col(8, ra, 'SINGLE', 4) &
        condition_item(citex4ds, climate_ref_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        condition_item(citex4ds, climate_ref_table, select, did, =, 'did of farm_data ', 'SHORT') &
```

```
        condition_item(citex4ds, climate_ref_table, select, month, =, 'month of farm_data ', 'SHORT')
&
        query_fs(citex4ds, climate_ref_table, ['All Fields']) &
        sql_select(climate_ref_table, ['SELECT', '*', 'FROM climate_ref_table WHERE','gid = ',
_78140,and,'did = ', _78661,and,'month = ', _79158]) :-
                farm_data :: get_value(gid(_78140)),
                farm_data :: get_value(did(_78661)),
                farm_data :: get_value(month(_79158)) &
        super(citex4ds)
}.
sector_table :: {
        tab(sector_table) &
        col(1, sid, 'SHORT', 2) &
        col(2, sname, 'TEXT', 50) &
        condition_item(citex4ds, sector_table, select, sid, =, 'sid of farm_data ', 'SHORT') &
        query_fs(citex4ds, sector_table, ['All Fields']) &
        sql_select(sector_table, ['SELECT', '*', 'FROM sector_table WHERE','sid = ', _8392]) :-
        farm_data :: get_value(sid(_8392)) &
        super(citex4ds)
}.
governorate_table :: {
        tab(governorate_table) &
        col(1, sid, 'SHORT', 2) &
        col(2, gid, 'SHORT', 2) &
        col(3, gname, 'TEXT', 50) &
        condition_item(citex4ds, governorate_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        query_fs(citex4ds, governorate_table, ['All Fields']) &
        sql_select(governorate_table, ['SELECT', '*', 'FROM governorate_table WHERE','gid = ',
_11888]) :-
                farm_data :: get_value(gid(_11888)) &
        super(citex4ds)
}.

directorate_table :: {
        tab(directorate_table) &
        col(1, sid, 'SHORT', 2) &
        col(2, gid, 'SHORT', 2) &
        col(3, did, 'SHORT', 2) &
        col(4, dname, 'TEXT', 50) &
        condition_item(citex4ds, directorate_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        condition_item(citex4ds, directorate_table, select, did, =, 'did of farm_data ', 'SHORT') &
        query_fs(citex4ds, directorate_table, ['All Fields']) &
        sql_select(directorate_table, ['SELECT', '*', 'FROM directorate_table WHERE','gid = ',
_15961,and,'did = ', _16458]) :-
                farm_data :: get_value(gid(_15961)),
                farm_data :: get_value(did(_16458)) &
        super(citex4ds)
}.
farm_table :: {
        tab(farm_table) &
        col(1, sid, 'SHORT', 2) &
        col(2, gid, 'SHORT', 2) &
        col(3, did, 'SHORT', 2) &
        col(4, fid, 'SHORT', 2) &
        col(5, fname, 'TEXT', 50) &
        col(6, area, 'SINGLE', 4) &
        col(7, plantation_date, 'DATE', 0) &
        col(8, irr_system, 'TEXT', 50) &
        col(9, fert_system, 'TEXT', 50) &
        col(10, drainage_system, 'TEXT', 50) &
```

```
		col(11, nt, 'SHORT', 2) &
		col(12, r_dist, 'SINGLE', 4) &
		col(13, t_dist, 'SINGLE', 4) &
		col(14, water_source, 'TEXT', 50) &
		col(15, user_cont_water, 'TEXT', 50) &
		col(16, variety_name, 'TEXT', 50) &
		col(17, s_s_month, 'SHORT', 2) &
		condition_item(citex4ds, farm_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
		condition_item(citex4ds, farm_table, select, did, =, 'did of farm_data ', 'SHORT') &
		condition_item(citex4ds, farm_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
		query_fs(citex4ds, farm_table, ['All Fields']) &
		sql_select(farm_table, ['SELECT', '*', 'FROM farm_table WHERE','gid = ', _27634,and,'did =
', _28155,and,'fid = ', _28652]) :-
			farm_data :: get_value(gid(_27634)),
			farm_data :: get_value(did(_28155)),
			farm_data :: get_value(fid(_28652)) &
		super(citex4ds)
}.
soil_table :: {
		tab(soil_table) &
		col(1, gid, 'SHORT', 2) &
		col(2, did, 'SHORT', 2) &
		col(3, fid, 'SHORT', 2) &
		col(4, texture, 'TEXT', 50) &
		col(5, water_table_level, 'SINGLE', 4) &
		col(6, ec, 'SINGLE', 4) &
		col(7, ph, 'SINGLE', 4) &
		col(8, fc, 'SINGLE', 4) &
		col(9, pwp, 'SINGLE', 4) &
		condition_item(citex4ds, soil_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
		condition_item(citex4ds, soil_table, select, did, =, 'did of farm_data ', 'SHORT') &
		condition_item(citex4ds, soil_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
		query_fs(citex4ds, soil_table, ['All Fields']) &
		sql_select(soil_table, ['SELECT', '*', 'FROM soil_table WHERE','gid = ', _35614,and,'did = ',
_36135,and,'fid = ', _36632]) :-
			farm_data :: get_value(gid(_35614)),
			farm_data :: get_value(did(_36135)),
			farm_data :: get_value(fid(_36632)) &
		super(soil_ref_table)
}.
climate_table :: {
		tab(climate_table) &
		col(1, gid, 'SHORT', 2) &
		col(2, did, 'SHORT', 2) &
		col(3, fid, 'SHORT', 2) &
		col(4, month, 'TEXT', 50) &
		col(5, avg_tc, 'SINGLE', 4) &
		col(6, avg_rh, 'SINGLE', 4) &
		col(7, ash, 'SINGLE', 4) &
		col(8, msh, 'SINGLE', 4) &
		col(9, ra, 'SINGLE', 4) &
		condition_item(citex4ds, climate_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
		condition_item(citex4ds, climate_table, select, did, =, 'did of farm_data ', 'SHORT') &
		condition_item(citex4ds, climate_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
		condition_item(citex4ds, climate_table, select, month, =, 'month of farm_data ', 'SHORT') &
		query_fs(citex4ds, climate_table, ['All Fields']) &
		sql_select(climate_table, ['SELECT', '*', 'FROM climate_table WHERE','gid = ',
_43686,and,'did = ', _44207,and,'fid = ', _44728,and,'month = ', _45225]) :-
			farm_data :: get_value(gid(_43686)),
			farm_data :: get_value(did(_44207)),
```

```
                farm_data :: get_value(fid(_44728)),
                farm_data :: get_value(month(_45225)) &
        super(climate_ref_table)
}.
water_table :: {
        tab(water_table) &
        col(1, gid, 'SHORT', 2) &
        col(2, did, 'SHORT', 2) &
        col(3, fid, 'SHORT', 2) &
        col(4, eciw, 'SINGLE', 4) &
        condition_item(citex4ds, water_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        condition_item(citex4ds, water_table, select, did, =, 'did of farm_data ', 'SHORT') &
        condition_item(citex4ds, water_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
        query_fs(citex4ds, water_table, ['All Fields']) &
        sql_select(water_table, ['SELECT', '*', 'FROM water_table WHERE','gid = ', _49788,and,'did
= ', _50309,and,'fid = ', _50806]) :-
                farm_data :: get_value(gid(_49788)),
                farm_data :: get_value(did(_50309)),
                farm_data :: get_value(fid(_50806)) &
        super(water_ref_table)
}.
soil_assessment_table :: {
        tab(soil_assessment_table) &
        col(1, gid, 'SHORT', 2) &
        col(2, did, 'SHORT', 2) &
        col(3, fid, 'SHORT', 2) &
        col(4, boron, 'SINGLE', 4) &
        col(5, chloride_sulphate, 'SINGLE', 4) &
        col(6, rsc, 'SINGLE', 4) &
        col(7, sar, 'SINGLE', 4) &
        col(8, profile_depth, 'SINGLE', 4) &
        col(9, ca_carbonate, 'SINGLE', 4) &
        col(10, max_d_tc_ss, 'SINGLE', 4) &
        col(11, min_d_rh_ss, 'SINGLE', 4) &
        col(12, esp, 'SINGLE', 4) &
        condition_item(citex4ds, soil_assessment_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        condition_item(citex4ds, soil_assessment_table, select, did, =, 'did of farm_data ', 'SHORT') &
        condition_item(citex4ds, soil_assessment_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
        query_fs(citex4ds, soil_assessment_table, ['All Fields']) &
        sql_select(soil_assessment_table, ['SELECT', '*', 'FROM soil_assessment_table WHERE','gid
= ', _59414,and,'did = ', _59935,and,'fid = ', _60432]) :-
                farm_data :: get_value(gid(_59414)),
                farm_data :: get_value(did(_59935)),
                farm_data :: get_value(fid(_60432)) &
        super(citex4ds)
}.
select_table :: {
        tab(select_table) &
        col(1, sid, 'SHORT', 2) &
        col(2, gid, 'SHORT', 2) &
        col(3, did, 'SHORT', 2) &
        col(4, fid, 'SHORT', 2) &

        condition_item(citex4ds, select_table, select, gid, =, 'gid of farm_data ', 'SHORT') &
        condition_item(citex4ds, select_table, select, did, =, 'did of farm_data ', 'SHORT') &
        condition_item(citex4ds, select_table, select, fid, =, 'fid of farm_data ', 'SHORT') &
        query_fs(citex4ds, select_table, ['All Fields']) &
        sql_select(select_table, ['SELECT', '*', 'FROM select_table']) &
        super(citex4ds)
}.
```

7. User Interface

```
/* File name : citex_diag_dlg.pl */
:- ensure_loaded('$KROL/lib/buttonbox').
:- ensure_loaded('$KROL/lib/ComboBox').
:- ensure_loaded('$KROL/lib/frame').
:- ensure_loaded('$KROL/lib/HList').
:- ensure_loaded('$KROL/lib/labelframe').
:- dynamic prop/2.
:- dynamic val/3.
:- dynamic finding/4.
:- dynamic prop_type/4.
citex_diag_dlg :-
        krol_init :: set(mode(un)),
        tcl :: eval(['proc get_disorders {args}',br([prolog,dq(get_disorders)])]),
        tcl :: eval(['proc show_properties {args}',br([prolog,dq(show_properties)])]),
        tcl :: eval(['proc show_values {args}',br([prolog,dq(show_values)])]),
        citex_diag_dlg :: run,
        init_disorders,
        citex_diag_dlg :: tkwait.

citex_diag_dlg :: {
widget(citex_diag_dlg, []) &
window_title(Title):-
        (appl_pdw :: get(sys(diag)) ->      (!,Title = 'Citex Diagnosis')
                ;
                Title = 'Citex Diagnosis & Treatment'
        ) &
components(Xs) :- self(D), :findall(X, D :: cs(_, X), Xs) &
pack(citex_diag_frm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_frm, citex_diag_dlg) &
pack(citex_diag_ses_all_lft_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_ses_all_lft_frm, citex_diag_frm) &
pack(citex_diag_ses_all_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_ses_all_frm, citex_diag_ses_all_lft_frm) &
pack(citex_diag_all_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_all_lblfrm, citex_diag_ses_all_frm) &
pack(citex_diag_all_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_all_hlst, citex_diag_all_lblfrm) &
pack(citex_diag_down_lft_btn, ['-side',bottom,'-fill',both,'-anchor',s]) &
c(citex_diag_down_lft_btn, citex_diag_all_lblfrm) &
pack(citex_diag_dwn_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_dwn_lblfrm, citex_diag_ses_all_frm) &
pack(citex_diag_dwn_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_dwn_hlst, citex_diag_dwn_lblfrm) &
pack(citex_diag_del_btn, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_del_btn, citex_diag_dwn_lblfrm) &
pack(citex_diag_left_btncitex_diag_dlg, ['-side',right,'-expand',true,'-fill',both,'-anchor',e]) &
c(citex_diag_left_btncitex_diag_dlg, citex_diag_ses_all_lft_frm) &
pack(citex_diag_conc_prop_val_fin_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_conc_prop_val_fin_frm, citex_diag_frm) &
pack(citex_diag_conc_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_conc_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_concept_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_concept_hlst, citex_diag_conc_lblfrm) &
pack(citex_diag_Property_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_Property_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_property_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_property_hlst, citex_diag_Property_lblfrm) &
```

```
pack(citex_diag_value_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_value_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_value_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_value_hlst, citex_diag_value_lblfrm) &
pack(citex_diag_down_btncitex_diag_dlg, ['-side',bottom,'-fill',both,'-anchor',s]) &
c(citex_diag_down_btncitex_diag_dlg, citex_diag_value_lblfrm) &
pack(citex_diag_finding_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_finding_lblfrm, citex_diag_conc_prop_val_fin_frm) &
pack(citex_diag_finding_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_finding_hlst, citex_diag_finding_lblfrm) &
pack(citex_diag_why_what_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',s]) &
c(citex_diag_why_what_btncitex_diag_dlg, citex_diag_finding_lblfrm) &
pack(citex_diag_rgt_sus_conf_hi_frm, ['-side',left,'-expand',true,'-fill',both,'-anchor',w]) &
c(citex_diag_rgt_sus_conf_hi_frm, citex_diag_frm) &
pack(citex_diag_3_rgt_btncitex_diag_dlg, ['-side',left,'-expand',true,'-fill',both,'-anchor',e]) &
c(citex_diag_3_rgt_btncitex_diag_dlg, citex_diag_rgt_sus_conf_hi_frm) &
pack(citex_diag_sus_conf_hi_frm, ['-side',right,'-expand',true,'-fill',both,'-anchor',e]) &
c(citex_diag_sus_conf_hi_frm, citex_diag_rgt_sus_conf_hi_frm) &
pack(citex_diag_sus_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_sus_lblfrm, citex_diag_sus_conf_hi_frm) &
pack(citex_diag_sus_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_sus_hlst, citex_diag_sus_lblfrm) &
pack(citex_diag_how_sus_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',s]) &
c(citex_diag_how_sus_btncitex_diag_dlg, citex_diag_sus_lblfrm) &
pack(citex_diag_confirm_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_confirm_lblfrm, citex_diag_sus_conf_hi_frm) &
pack(citex_diag_conf_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_conf_hlst, citex_diag_confirm_lblfrm) &
pack(citex_diag_how_conf_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',s]) &
c(citex_diag_how_conf_btncitex_diag_dlg, citex_diag_confirm_lblfrm) &
pack(citex_diag_hi_lblfrm, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_hi_lblfrm, citex_diag_sus_conf_hi_frm) &
pack(citex_diag_hi_hlst, ['-side',top,'-expand',true,'-fill',both,'-anchor',n]) &
c(citex_diag_hi_hlst, citex_diag_hi_lblfrm) &
pack(citex_diag_how_hi_btncitex_diag_dlg, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',s]) &
c(citex_diag_how_hi_btncitex_diag_dlg, citex_diag_hi_lblfrm) &
pack(T, ['-side',bottom,'-expand',true,'-fill',both,'-anchor',s]) :-
        (appl_pdw :: get(sys(diag)) ->      (!,T = citex_diag_ok_cancel_btncitex_diag_dlg)
                ;
                T = citex_diagTreat_ok_cancel_btncitex_diag_dlg
        ) &
c(T, citex_diag_dlg) :-
        (appl_pdw :: get(sys(diag)) ->      (!,T = citex_diag_ok_cancel_btncitex_diag_dlg)
                ;
                T = citex_diagTreat_ok_cancel_btncitex_diag_dlg
        ) &

super(dialog)
}.
citex_diag_3_rgt_btncitex_diag_dlg :: {
widget(citex_diag_3_rgt_btncitex_diag_dlg, ['-orient',vertical], ['-padx','','-pady',"]) &
default(move_to_sus) &
button(move_to_sus, ['-bg', gray,'-image','Arrowrt.gif','-command','citex_diag_3_rgt_btncitex_diag_dlg
:: move_sus'], '') &
% display the suspected disorder
move_sus :-

        :findall(C-P-V,finding(_,C,P,V),Fin),
        (       Fin = [] ->
                krol_msgs :: show("There is no finding  ....",[])
```

```
        ;       :set_susbs(Fin,s),
                disorder :: reset_att(suspected/1),
                diag_inference :: predict,
                disorder :: get(suspected(Dsu)),
                (       Dsu = [] ->
                        krol_msgs :: show("There is no sufficient finidings to suspect disorders
....",[])
                ;
                        :sort(Dsu, Dsu1),
                        citex_diag_sus_hlst :: clean(citex_diag_dlg),
                        :insert_in_hlist(Dsu1,citex_diag_sus_hlst),
                        :in_process(Dsu1, confirmed, IPs),
                        confirm_disorders :: abduct_all(IPs, OPs),
                        :retractall(prop(_,_)),
                        :retractall(val(_,_,_)),
                        :out_process(OPs,Cps),
                        :sort(Cps,Cps1),
                        citex_diag_property_hlst :: clean(citex_diag_dlg),
                        citex_diag_value_hlst :: clean(citex_diag_dlg),
                        citex_diag_conf_hlst :: clean(citex_diag_dlg),
                        citex_diag_hi_hlst :: clean(citex_diag_dlg),
                        :insert_in_hlist(Cps1,citex_diag_concept_hlst)
                )
        ) &

% display the confirm disorder
button(move_to_confirm, ['-bg', gray,'-image','Arrowrt.gif','-
command','citex_diag_3_rgt_btncitex_diag_dlg :: move_to_confirm'], '') &
move_to_confirm :-
        :findall(C-P-V,finding(_,C,P,V),Fin),
        (       Fin = [] ->
                krol_msgs :: show("There is no finding  ....",[])
        ;       disorder :: get(suspected(L)),
                (       L = [] ->
                        krol_msgs :: show("Please, get suspected disoredrs.. first  ....",[])
                ;       :set_susbs(Fin,c),
                        disorder :: reset_att(suspected/1),
                        :set_sus_dis(L,suspected),
                        disorder :: reset_att(confirmed/1),
                        diag_inference :: confirm,
                        disorder :: get(confirmed(Dsu)),
                        (       Dsu = [] ->
                                krol_msgs :: show("There is no sufficient finidings to confirm
disorders  ....",[])
                        ;       :sort(Dsu, Dsu1),
                                citex_diag_conf_hlst :: clean(citex_diag_dlg),
                                :insert_in_hlist(Dsu1,citex_diag_conf_hlst),
                                :in_process(Dsu1, highly_confirmed, IPs),
                                verify_disorders :: abduct_all(IPs, OPs),
                                :retractall(prop(_,_)),
                                :retractall(val(_,_,_)),
                                :out_process(OPs,Cps),
                                :sort(Cps,Cps1),
                                citex_diag_property_hlst :: clean(citex_diag_dlg),
                                citex_diag_value_hlst :: clean(citex_diag_dlg),
                                citex_diag_hi_hlst :: clean(citex_diag_dlg),
                                :insert_in_hlist(Cps1,citex_diag_concept_hlst)
                        )
                )
        )&
```

```
% display the highy confirm disorder
button(move_to_hiconfirm, ['-bg', gray,'-image','Arrowrt.gif','-
command','citex_diag_3_rgt_btncitex_diag_dlg :: move_to_hiconfirm'], '') &
move_to_hiconfirm :-
        :findall(C-P-V,finding(_,C,P,V),Fin),
        (       Fin = [] ->
                krol_msgs :: show("There is no finding  ....",[])
        ;       disorder :: get(confirmed(L)),
                (       L = [] ->
                        krol_msgs :: show("Please, get confirmed disoredrs.. first  ....",[])
                ;       :set_susbs(Fin,h),
                        disorder :: reset_att(confirmed/1),
                        :set_sus_dis(L,confirmed),
                        disorder :: reset_att(highly_confirmed/1),
                        diag_inference :: verify,
                        disorder :: get(highly_confirmed(Dcu)),
                        (       Dcu = [] ->
                                krol_msgs :: show("There is no sufficient finidings to
get on highly confirmed disorders  ....",[])
                        ;       :sort(Dcu, Dcu1),
                                citex_diag_hi_hlst :: clean(citex_diag_dlg),
                                :insert_in_hlist(Dcu1,citex_diag_hi_hlst),
                                citex_diag_property_hlst :: clean(citex_diag_dlg),
                                citex_diag_value_hlst :: clean(citex_diag_dlg),
                                citex_diag_concept_hlst :: clean(citex_diag_dlg)
                        )
                )
        )&

super(buttonbox)
}.
citex_diag_Property_lblfrm :: {
widget(citex_diag_Property_lblfrm, ['-label','Properties','-labelside',top], []) &
super(labelframe)
}.
citex_diag_all_lblfrm :: {
widget(citex_diag_all_lblfrm, ['-label','All Disorder','-labelside',top], []) &
super(labelframe)
}.
citex_diag_dwn_lblfrm :: {
widget(citex_diag_dwn_lblfrm, ['-label','User Suspected Disorder','-labelside',top], []) &
super(labelframe)
}.
citex_diag_conc_lblfrm :: {
widget(citex_diag_conc_lblfrm, ['-label','Concepts','-labelside',top], []) &
super(labelframe)
}.
citex_diag_conc_prop_val_fin_frm :: {
widget(citex_diag_conc_prop_val_fin_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.
citex_diag_confirm_lblfrm :: {
widget(citex_diag_confirm_lblfrm, ['-label','Confirmed Disorders','-labelside',top], []) &
super(labelframe)
}.
citex_diag_down_btncitex_diag_dlg :: {
widget(citex_diag_down_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady',"]) &
default(move_down) &
button(move_down, ['-bg', gray,'-image','Arrowdwn.gif','-
command','citex_diag_down_btncitex_diag_dlg :: move_down'], '') &
```

```
% move the legal value to the button list
move_down :-
        citex_diag_concept_hlst :: fetch(citex_diag_dlg,C),
        citex_diag_property_hlst :: fetch(citex_diag_dlg,P),
        citex_diag_value_hlst :: fetch(citex_diag_dlg,V),
        (       (C = '/' ; P = '/' ; V = '/') ->
                :true
        ;       :format_to_chars('~q of ~q = ~q', [P,C,V], Str),
                :name(I, Str),
                (       citex_diag_finding_hlst :: item(I) ->
                        :true
                ;       citex_diag_finding_hlst :: insert_item(citex_diag_dlg,I),
                        :assert(finding(I,C,P,V))
                )
        ) &
super(buttonbox)
}.
%%%%%%%%%%%%%%%%%%%%
citex_diag_down_lft_btn :: {
widget(citex_diag_down_lft_btn, ['-orient',horizontal], ['-padx','','-pady',"]) &
default(move_down) &
button(move_down, ['-bg', gray,'-image','Arrowdwn.gif','-command','citex_diag_down_lft_btn ::
move_to_user_sus'], '') &
% move the disorder to the user suspected list
move_to_user_sus :-
        citex_diag_all_hlst :: fetch(citex_diag_dlg,C),
        (       (C = '/' ) ->
                :true
        ;       citex_diag_dwn_hlst :: is_item(citex_diag_dlg,C) ->
                :true
        ;       citex_diag_dwn_hlst :: insert_item(citex_diag_dlg,C)
        ) &

super(buttonbox)
}.
citex_diag_finding_lblfrm :: {
widget(citex_diag_finding_lblfrm, ['-label','Findings','-labelside',top], []) &
super(labelframe)
}.

citex_diag_frm :: {
widget(citex_diag_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.

citex_diag_hi_lblfrm :: {
widget(citex_diag_hi_lblfrm, ['-label','High Confirmed Disorders','-labelside',top], []) &
super(labelframe)
}.

citex_diag_how_conf_btncitex_diag_dlg :: {
widget(citex_diag_how_conf_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady',"]) &
        default(how_conf) &
button(how_conf, ['-text','How','-command','citex_diag_how_conf_btncitex_diag_dlg :: how_conf'], '')
&
how_conf :-
        (
        krol_msgs :: show("It will be implemented soon....",[])
        )&
super(buttonbox)
```

```
}.

citex_diag_how_hi_btncitex_diag_dlg :: {
widget(citex_diag_how_hi_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady',"]) &
default(how_hi) &
button(how_hi, ['-text','How','-command','citex_diag_how_hi_btncitex_diag_dlg :: how_hi'], '') &
how_hi :-
        krol_msgs :: show("It will be implemented soon....",[])&
super(buttonbox)
}.

citex_diag_how_sus_btncitex_diag_dlg :: {
widget(citex_diag_how_sus_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady',"]) &
default(how_sus) &
button(how_sus, ['-text','How','-command','citex_diag_how_sus_btncitex_diag_dlg :: how_sus'], '') &
how_sus :-
        krol_msgs :: show("It will be implemented soon....",[])&

super(buttonbox)
}.

citex_diag_left_btncitex_diag_dlg :: {
widget(citex_diag_left_btncitex_diag_dlg, ['-orient',vertical], ['-padx','','-pady',"]) &
default(move_left) &
button(move_left, ['-bg', gray,'-image','Arrowrt.gif','-command','citex_diag_left_btncitex_diag_dlg ::
move_left'], '') &

%maryam   minimize the list of observation
move_left :-
        citex_diag_dwn_hlst :: content(citex_diag_dlg, L1),
        (       L1 = [] ->
                :true
        ;       :retractall(prop(_,_)),
                :retractall(val(_,_,_)),
                :retractall(finding(_,_,_,_)),
                :retractall(prop_type(_,_,_,_)),
                :in_process(L1, suspected, IPs),
                caused_by_disoders :: abduct_all(IPs, OPs),
                :out_process(OPs,Cps),
                :sort(Cps,Cps1),
                citex_diag_property_hlst :: clean(citex_diag_dlg),
                citex_diag_value_hlst :: clean(citex_diag_dlg),
                citex_diag_finding_hlst :: clean(citex_diag_dlg),
                citex_diag_sus_hlst :: clean(citex_diag_dlg),
                citex_diag_conf_hlst :: clean(citex_diag_dlg),
                citex_diag_hi_hlst :: clean(citex_diag_dlg),
                :insert_in_hlist(Cps1,citex_diag_concept_hlst)
        )&

super(buttonbox)
}.

citex_diag_ok_cancel_btncitex_diag_dlg :: {
widget(citex_diag_ok_cancel_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady',"]) &
default(ok) &
button(ok, ['-text','Ok','-command','citex_diag_ok_cancel_btncitex_diag_dlg :: ok'], '') &
ok :-
        citex_diag_dlg :: destroy &

super(buttonbox)
```

```
}.

citex_diagTreat_ok_cancel_btncitex_diag_dlg :: {
widget(citex_diagTreat_ok_cancel_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady',"]) &
default(ok) &
button(ok, ['-text','Ok','-command','citex_diagTreat_ok_cancel_btncitex_diag_dlg :: ok'], '') &
ok :-
        citex_diag_dlg :: destroy &

button(treat, ['-text','Treatment','-command','citex_diagTreat_ok_cancel_btncitex_diag_dlg :: treat'], '')
&
treat :-
% Maryam confirm, high confirm lists
        disorder :: get(confirmed(L1)),
        disorder :: get(highly_confirmed(L2)),
        :append(L1,L2,Dis),
        (       Dis = [] ->
                krol_msgs :: show("There are no disorders confirmed",[])
        ;
                disorder :: reset_att(confirmed/1),
                disorder :: reset_att(highly_confirmed/1),
                krol_init :: init,
                :set_sus_dis(L1,confirmed),
                :set_sus_dis(L2,highly_confirmed),
                krol_init :: set(mode(cm)),
                treat_task_uncondional :: start_inference
        ) &

super(buttonbox)
}.

citex_diag_rgt_sus_conf_hi_frm :: {
widget(citex_diag_rgt_sus_conf_hi_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.

citex_diag_ses_all_frm :: {
widget(citex_diag_ses_all_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.

citex_diag_ses_all_lft_frm :: {
widget(citex_diag_ses_all_lft_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.

citex_diag_sus_conf_hi_frm :: {
widget(citex_diag_sus_conf_hi_frm, ['-width',0,'-height',0,'-borderwidth',0], []) &
super(frame)
}.

citex_diag_sus_lblfrm :: {
widget(citex_diag_sus_lblfrm, ['-label','Suspected Disorders','-labelside',top], []) &
super(labelframe)
}.

citex_diag_all_hlst :: {
widget(citex_diag_all_hlst, ['-scrollbar', auto], ['hlist.selectmode multiple','hlist.itemtype imagetext
hlist.drawBranch false hlist.indent 14 hlist.wideSelect false']) &
super(hlist)
```

```
}.

citex_diag_dwn_hlst :: {
widget(citex_diag_dwn_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false']) &
super(hlist)
}.


citex_diag_concept_hlst :: {
widget(citex_diag_concept_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
configure(['-browsecmd show_properties']) &
super(hlist)
}.

citex_diag_property_hlst :: {
widget(citex_diag_property_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
configure(['-browsecmd show_values']) &
super(hlist)
}.

citex_diag_value_hlst :: {
widget(citex_diag_value_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
super(hlist)
}.

citex_diag_finding_hlst :: {
widget(citex_diag_finding_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false hlist.height 4']) &
super(hlist)
}.

citex_diag_sus_hlst :: {
widget(citex_diag_sus_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false hlist.height 5']) &
super(hlist)
}.

citex_diag_conf_hlst :: {
widget(citex_diag_conf_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false hlist.height 5']) &
super(hlist)
}.

citex_diag_hi_hlst :: {
widget(citex_diag_hi_hlst, ['-scrollbar', auto], ['hlist.itemtype imagetext hlist.drawBranch false
hlist.indent 14 hlist.wideSelect false hlist.height 5']) &
super(hlist)
}.

citex_diag_value_lblfrm :: {
widget(citex_diag_value_lblfrm, ['-label','Values','-labelside',top], []) &
super(labelframe)
}.

citex_diag_why_what_btncitex_diag_dlg :: {
widget(citex_diag_why_what_btncitex_diag_dlg, ['-orient',horizontal], ['-padx','','-pady',"]) &
```

```
default(why) &
button(why, ['-text','Why','-command','citex_diag_why_what_btncitex_diag_dlg :: why'], '') &
why :-
        krol_msgs :: show("It will be implemented soon....",[])&

button(delete, ['-text','Delete','-command','citex_diag_why_what_btncitex_diag_dlg :: delete'], '') &
delete :-
        citex_diag_finding_hlst :: fetch(citex_diag_dlg,I),
        (       I = '/' ->
                :true
        ;       citex_diag_finding_hlst :: delete_item(citex_diag_dlg,I),
                :finding(I,C,P,V),
                :retractall(finding(I,_C,_P,_V)),
                :delete_subs(C,P,V)
        ) &

button(what, ['-text','What','-command','citex_diag_why_what_btncitex_diag_dlg :: what'], '') &
what :-
        krol_msgs :: show("It will be implemented soon....",[])&

super(buttonbox)
}.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

citex_diag_del_btn :: {
widget(citex_diag_del_btn, ['-orient',horizontal], ['-padx','','-pady','']) &
default(del) &
button(delete, ['-text','Delete','-command','citex_diag_del_btn :: delete'], '') &
delete :-
        citex_diag_dwn_hlst :: fetch(citex_diag_dlg,I),
        (       I = '/' ->
                :true
        ;       citex_diag_dwn_hlst :: delete_item(citex_diag_dlg,I),
                :retractall(prop(_,_)),
                :retractall(val(_,_,_)),
                :retractall(finding(_,_,_,_)),
                :retractall(prop_type(_,_,_,_)),
                citex_diag_concept_hlst :: clean(citex_diag_dlg),
                citex_diag_property_hlst :: clean(citex_diag_dlg),
                citex_diag_value_hlst :: clean(citex_diag_dlg),
                citex_diag_finding_hlst :: clean(citex_diag_dlg),
                citex_diag_sus_hlst :: clean(citex_diag_dlg),
                citex_diag_conf_hlst :: clean(citex_diag_dlg),
                citex_diag_hi_hlst :: clean(citex_diag_dlg),
                :findall(X,citex_diag_dwn_hlst :: is_item(citex_diag_dlg,X),L1),
                (       L1 = [] ->
                        :true
                ;       :in_process(L1, suspected, IPs),
                        caused_by_disoders :: abduct_all(IPs, OPs),
                        :out_process(OPs,Cps),
                        :sort(Cps,Cps1),
                        :insert_in_hlist(Cps1,citex_diag_concept_hlst)
                )
        )&


super(buttonbox)
}.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
get_disorders :-
```

```
        krol_init :: init,
        plant :: get_value(season(Season)),
        season(Season,L),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        retractall(finding(_,_,_,_)),
        retractall(prop_type(_,_,_,_)),
        sort(L,L1),
        insert_in_hlist(L1,citex_diag_all_hlst),
        in_process(L1, suspected, IPs),
        caused_by_disoders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_finding_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        citex_diag_dwn_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).

init_disorders :-
        krol_init :: init,
        plant :: get_value(season(Seson)),
        season(Seson,L),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        retractall(finding(_,_,_,_)),
        retractall(prop_type(_,_,_,_)),
        sort(L,L1),
        insert_in_hlist(L1,citex_diag_all_hlst),
        in_process(L1, suspected, IPs),
        caused_by_disoders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_finding_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        citex_diag_dwn_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).

show_properties:-
        citex_diag_concept_hlst :: fetch(citex_diag_dlg,C),
        findall(P,prop(C,P),Lp),
        sort(Lp, Lp1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Lp1,citex_diag_property_hlst).

show_values:-
        citex_diag_concept_hlst :: fetch(citex_diag_dlg,C),
        citex_diag_property_hlst :: fetch(citex_diag_dlg,P),
        findall(V,val(C,P,V),Lv),
        sort(Lv, Lv1),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Lv1,citex_diag_value_hlst).
```

```
clean_hlist(H) :-
        H :: clean(citex_diag_dlg).

insert_in_hlist(L,H) :-
        H :: clean(citex_diag_dlg),
        H :: insert(citex_diag_dlg,L).

insert_in_hlist1([],_).
insert_in_hlist1([H|T],O) :-
        O :: insert_item(citex_diag_dlg,H),
        insert_in_hlist1(T,O).

in_process([], _, []).
in_process([D|Ds], F, [P in disorder|IPs]) :-
        P =.. [F,D],
        in_process(Ds, F, IPs).

out_process([], []).
out_process([C-P-V|OPs], [C|Cs]) :-
        (       prop(C,P) ->
                true
        ;       assert(prop(C,P))
        ),
        assert(val(C,P,V)),
        out_process(OPs, Cs).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
set_susbs([],_).
set_susbs([C-P-V|Vals],Type) :-
        (       prop_type(C,P,V,_) ->
                true
        ;       F =.. [P,V],
                C :: set_value(F),
                assert(prop_type(C,P,V,Type))
        ),
        set_susbs(Vals,Type).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
set_sus_dis([],_).
set_sus_dis([H|L],V) :-
        F =.. [V,H],
        disorder :: set_value(F),
        set_sus_dis(L,V).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
del_sus_dis([]).
del_sus_dis([H|L]) :-
        H :: reset,
        del_sus_dis(L).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
delete_subs(C,A,V) :-
        (       C :: is_single(A/1) ->
                C :: reset_att(A/1)
        ;       P =.. [A,Vs],
                C :: get(P),
                delete(Vs, V, Vs1),
                P1 =.. [A, Vs1],
```

```
                C :: set(P1)
        ),
        retract(prop_type(C,A,V,Type)),
        handle_p_type(Type).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%handle_p_type(h)
:-
        disorder :: get(suspected(L)),
        disorder :: reset_att(suspected/1),
        set_sus_dis(L,suspected),
        disorder :: reset_att(confirmed/1),
        diag_inference :: confirm ,
        disorder :: get(confirmed(Dsu)),
        sort(Dsu, Dsu1),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Dsu1,citex_diag_conf_hlst),
        in_process(Dsu1, highly_confirmed, IPs),
        verify_disorders :: abduct_all(IPs, OPs),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).
handle_p_type(c) :-
        handle_type(h),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_conf_hlst :: clean(citex_diag_dlg),
        citex_diag_hi_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
        disorder :: reset_att(suspected/1),
        diag_inference :: predict,
        disorder :: get(suspected(Dsu)),
        sort(Dsu, Dsu1),
        insert_in_hlist(Dsu1,citex_diag_sus_hlst),
        in_process(Dsu1, confirmed, IPs),
        confirm_disorders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        insert_in_hlist(Cps1,citex_diag_concept_hlst).
handle_p_type(s) :-
        handle_type(h),
        handle_type(c),
        plant :: get(season(Seson)),
        season(Seson,L),
        retractall(prop(_,_)),
        retractall(val(_,_,_)),
        in_process(L, suspected, IPs),
        caused_by_disoders :: abduct_all(IPs, OPs),
        out_process(OPs,Cps),
        sort(Cps,Cps1),
        citex_diag_property_hlst :: clean(citex_diag_dlg),
        citex_diag_value_hlst :: clean(citex_diag_dlg),
        citex_diag_sus_hlst :: clean(citex_diag_dlg),
```

```
	citex_diag_conf_hlst :: clean(citex_diag_dlg),
	citex_diag_hi_hlst :: clean(citex_diag_dlg),
	insert_in_hlist(Cps1,citex_diag_concept_hlst).

handle_type(X) :-
	forall(retract(prop_type(C,A,V,X)),
		(	retract(finding(I,C,A,V)),
			citex_diag_finding_hlst :: delete_item(citex_diag_dlg,I),
			(	C :: is_single(A/1) ->
				C :: reset_att(A/1)
			;	P =.. [A,Vs],
				C :: get(P),
				delete(Vs, V, Vs1),
				P1 =.. [A, Vs1],
				C :: set(P1)
			)
		)
	).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
get_treat(Ds) :-
	sort(Ds,Ds1),
	get_treat(Ds1, Ts1),
	sort(date_sort) :: qsort(Ts1, Ts),
	treat_dialog :: display,
	show_treat(Ts),
	treat_dialog :: tkwait.

get_treat([], []).
get_treat([D|Ds], Ts) :-
	D :: get(method(Method)),
	(	Method = [] ->
		D :: leaves(Ls),
		(	(Ls = []; Ls = [D]) ->
			format_to_chars("~tThere is no Treatment for the disease
~w~n~n",[D],Treat1),
			name(Treat, Treat1),
			T = [Treat]
		;	append(Ls, Ds, Ds1),
			get_treat(Ds1, Ts)
		)
	;	my_get(D, material_name(Matx), operation),
		sort(Matx, Mat),
		my_get(D, number(Number), treat_op),
		my_get(D, date(Datex), treat_op),
		(	Datex = [] ->
			my_get(D, special_date(Date), treat_op)
		;	Date = Datex
		),
		my_get(D, material_qty(Qty), operation),
		my_get(D, unit(Unit), operation),
		(	(Method = painting ; Method = disinfection ; Method = 'soil treatment') ->
			AT = 'any suitable time'
		;	(Method = 'chemical spray' ; Method = 'foliage nutrition') ->
			AT = 'early morning or afternoon'
		;	AT = ''
		),
		my_get(D, advice(Advx), treat_op),
		sort(Advx, Adv),
```

```
                (       (Mat = [], Method = [], Number = [], Date = [], Qty = [], Unit = [], AT = [],
Adv = []) ->
                        format_to_chars("~tThere is no Treatment for the disease
~w~n~n",[D],Treat1),
                        name(Treat, Treat1),
                        T = [Treat]
                ;       format_to_chars("Treatment of disorder ~w is :~n",[D],D1),
                        format_to_chars("       Material : ~w~n",[Mat],Mat1),
                        format_to_chars("       Method : ~w~n",[Method], Method1),
                        format_to_chars("       Number : ~w~n",[Number], Number1),
                        format_to_chars("       Qty : ~w~n",[Qty], Qty1),
                        format_to_chars("       Unit : ~w~n",[Unit], Unit1),
                        format_to_chars("       Application Time : ~w~n",[AT], AT1),
                        format_to_chars("       Advice : ~w~n~n",[Adv], Adv1),
                        name(D11, D1),
                        name(Mat11, Mat1),
                        name(Method11, Method1),
                        name(Number11, Number1),
                        name(Qty11, Qty1),
                        name(Unit11, Unit1),
                        name(AT11, AT1),
                        name(Adv11, Adv1),
                        T = [D11,Mat11,Method11,Number11,Date,Qty11,Unit11,AT11,Adv11]
                ),
                Ts = [T|Ts1]
        ),
        get_treat(Ds, Ts1).

my_get(D, P, Super) :-
        D = Super, !,
        D :: get(P).

my_get(D, P, Super) :-
        copy_term(P, P1),
        D :: get(P1),
        (       arg(1, P1, []) ->
                D :: super(S),
                my_get(S, P, Super)
        ;       P = P1
        ).

/* File name : treat_dlg.pl */
:- ensure_loaded('$KROL/lib/flatten').
:- ensure_loaded('$KROL/lib/txtw').
:- ensure_loaded('$KROL/lib/buttonbox').
:- use_module(library(lists), [prefix/2]).

treat_dialog :: {
belong_to(citex_diag_dlg) &

window_title('Treatment Result') &

widget(treat_dialog, []) &

components([
                treat_txt,
                treat_txt_buttons
]) &

handle_abnormal_exit :-
```

```
                treat_txt_buttons :: action(end) &

super(dialog)
}.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
treat_txt :: {
belong_to(treat_dialog) &

widget(treat_txt, ['-height', 480, '-width', 640], ['text.font 8x13']) &
pack(['-expand true -fill both']) &

super(textwindow)
}.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
treat_txt_buttons :: {
belong_to(treat_dialog) &

widget(treat_txt_buttons, Args, Options) :-
                Args = ['-orient horizontal'],
                Options = [] &

pack(['-fill x']) &

button(save, Args, Bind) :-
                Args = ['-text', 'Save', '-command', 'treat_txt_buttons :: action(save)',
                        '-underline 0', '-width 10'],
                Bind = '<Control-s>' &
button(close, Args, Bind) :-
                Args = ['-text', 'Close', '-command', 'treat_txt_buttons :: action(close)',
                        '-underline 0', '-width 10'],
                Bind = '<Control-e>' &

default(close) &

action(close) :-
                treat_dialog :: destroy &

action(save) :-
                tcl :: get_save_file('', File, 'Save Treatment Result File'),
                (       File = '' ->
                        :true
                ;       treat_txt :: fetch(T),
                        :open(File, write, Stream),
                        :format(Stream,'~w', [T]),
                        :close(Stream)
                ) &

super(buttonbox)
}.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

show_treat([]).
show_treat([X|Xs]) :-
                show_treat1(X),
                show_treat(Xs).

show_treat1([]).
show_treat1([X|Xs]) :-
```

```
(     (X = [D,M,Y], valid_date(M, D, Y)) ->
      formate_date(X1, X),
      format_to_chars("            Date : ~s~n",[X1], X11),
      name(X2, X11)
;     name(X, Y),
      (           prefix("next ", Y) ->
                  format_to_chars("            Date : ~s~n",[Y], X11),
                  name(X2, X11)
      ;           X2 = X
      )
),
treat_txt :: insert(X2),
show_treat1(Xs).
```

```
/* File name : season_dis.pl */
season(spring,[rose_scarab,citrus_flower_moth,psorosis,anthracnose,gummosis,wilt_root_rot,ganoder
ma_rot,armillaria_root_rot,alternaria_leaves_spot,gum_spots,lichens,citrus_white_fly,
scales,aphids,mealy_bug,leafminer,rust_mite,bud_mite,
brown_mite,flat_mite,citrus_nematude,nitrogen_def,phosphorus_def,potassium_def,magnesium_def,m
anganese_def,iron_def,calcium_def,zinc_def,salt_injury]).

season(summer,[psorosis,anthracnose,gummosis,wilt_root_rot,ganoderma_rot,armillaria_root_rot,alter
naria_leaves_spot,gum_spots,lichens,citrus_white_fly,scales,aphids,mealy_bug,leafminer,rust_mite,bu
d_mite,brown_mite,flat_mite,citrus_nematude,nitrogen_def,phosphorus_def,potassium_def,magnesium
_def,manganese_def,iron_def,calcium_def,zinc_def,salt_injury]).

season(autumn,[impieetratura,stubborn,sooty_mold,alternaria_rot,sun_burn,fruit_cracking,fruit_creasin
g,mediterranean_fruit_fly,green_stink_bug,psorosis,anthracnose,gummosis,wilt_root_rot,ganoderma_r
ot,armillaria_root_rot,alternaria_leaves_spot,gum_spots,lichens,citrus_white_fly,scales,aphids,mealy_b
ug,leafminer,rust_mite,bud_mite,brown_mite,flat_mite,citrus_nematude,nitrogen_def,phosphorus_def,
potassium_def, magnesium_def,manganese_def,iron_def,calcium_def,zinc_def,salt_injury]).

season(winter,[impieetratura,stubborn,sooty_mold,alternaria_rot,sun_burn,fruit_cracking,fruit_creasing
,mediterranean_fruit_fly,green_stink_bug,psorosis,anthracnose,gummosis,wilt_root_rot,ganoderma_rot
,armillaria_root_rot,alternaria_leaves_spot,gum_spots,lichens,citrus_white_fly,scales,aphids,mealy_bu
g,leafminer,rust_mite,bud_mite,brown_mite,flat_mite,citrus_nematude,nitrogen_def,phosphorus_def,p
otassium_def,magnesium_def,manganese_def,iron_def,calcium_def,zinc_def,salt_injury]).
```

8. Main interface

```
/* File name : main.pl */
:- use_module(library(ordsets), [ord_subtract/3]).
:- use_module(library(charsio), [format_to_chars/3,read_from_chars/2]).
:- use_module(library(system), [delete_file/1,file_exists/1, exec/3,
                        make_directory/1, working_directory/2, system/1, environ/2]).
:- ensure_loaded('$KROL/lib/menubar').
:- ensure_loaded('$KROL/lib/directory').
:- ensure_loaded([
          '$KROL/lib/date',
          '$KROL/lib/log',
          '$KROL/lib/krol_init',
          '$KROL/lib/stack',
          '$KROL/lib/msgs',
          '$KROL/lib/tk_user',
          '$KROL/lib/back_dlg',
          '$KROL/lib/database',
          '$KROL/lib/history',
          '$KROL/lib/gt',
          '$KROL/lib/rule_exp',
          '$KROL/lib/inferenc',
```

```
        '$KROL/lib/tab',
        '$KROL/lib/fun'
]).
:- ensure_loaded('diag_system').

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
appl_pdw :: {
attributes([
% Control fllags
        finding(0),addc(0), disorderc(0),cmf(0), newf(0), node(0), server(0),
% Counters
        dc(0), sc(0),
% Subsystem Flage
        sys([])
]) &

window_title('Citex Expert System') &

widget(gttpdm, []) &

geometry('400x300+100+100') &

menubotton(Widget,Txt,0,Bal,Status) :-
        Widget = db,
        Txt = 'Data Base',
        Bal = 'Data Base\\nMenu',
        Status = 'User' &

menu(db, [Label,0,Comm,Acc,Acc1], command) :-
        Label = 'User',
        Comm = 'userdatabase',
        Acc = 'Ctrl+u',
        Acc1 = '<Control-U>' &

menubotton(Widget,Txt,0,Bal,Status) :-
        Widget = expert_system,
        Txt = 'Expert System',
        Bal = 'Expert System\\nMenu',
        Status = 'Diagnosis , Treatment' &

menu(expert_system, [Label,Underline,Comm,Acc,Acc1], command) :-
        Label = 'Diagnosis ',
        Underline = 0,
        Comm = 'diagnosis',
        Acc = 'Ctrl+d',
        Acc1 = '<Control-D>' &

menu(expert_system, [Label,0,Comm,Acc,Acc1], command) :-
        Label = 'Treatment ',
        Comm = 'treatment',
        Acc = 'Ctrl+t',
        Acc1 = '<Control-T>' &


menubotton(Widget,Txt,0,Bal,Status) :-
        Widget = exit,
        Txt = 'Exit',
        Bal = 'Exit\\nMenu',
        Status = 'Exit' &
```

```
menu(exit, [Label,1,Comm,Acc,Acc1], command) :-
        Label = 'Exit ',
        Comm = 'exit',
        Acc = 'Ctrl+x',
        Acc1 = '<Control-x>' &

status(toolbar, S) :-
        tcl :: eval('set tbar', S1),
        :name(S, S1) &

super(pdwmenu)
}.

main:-
        tcl :: init,
        appl_pdw :: display,
        tcl :: end.

userdatabase:-
        exec('CitexDb.exe', [null,null,null], _).
diagnosis:-
        appl_pdw :: set(sys(diag)),diag_main.

treatment:-
        appl_pdw :: set(sys(treat)),treat_main.

exit:-
        appl_pdw :: destroy.

/* File name : diag_system.pl */
:-use_module(library(system)).
:-ensure_loaded('$KROL/lib/messages').
:-ensure_loaded('$KROL/lib/database').
:-ensure_loaded('$KROL/lib/tk_user').
:-ensure_loaded('$KROL/lib/date').
:-ensure_loaded(c_concept).
:-ensure_loaded('citex4.pl').
:-ensure_loaded(diag_rules).
:-ensure_loaded(diag_table).
:-ensure_loaded(season_dis).
:-ensure_loaded(citex_diag_dlg).
:-ensure_loaded(diag_task).
:-ensure_loaded(treat_task).
:-ensure_loaded(treat_rules).
:-ensure_loaded(treat_dlg).
:-ensure_loaded(order).
:-ensure_loaded(treat_inference).
:-ensure_loaded(diag_inference).

diag_start :-
            krol_init :: init,
            citex4ds :: open,
            select_table :: fetch([[SN,GN,DN,FN]]),
            farm_data :: set(sid(SN)),
            farm_data :: set(gid(GN)),
            farm_data :: set(did(DN)),
            farm_data :: set(fid(FN)),
            diag_task_uncondional :: start_inference,
            citex4ds :: close.
```

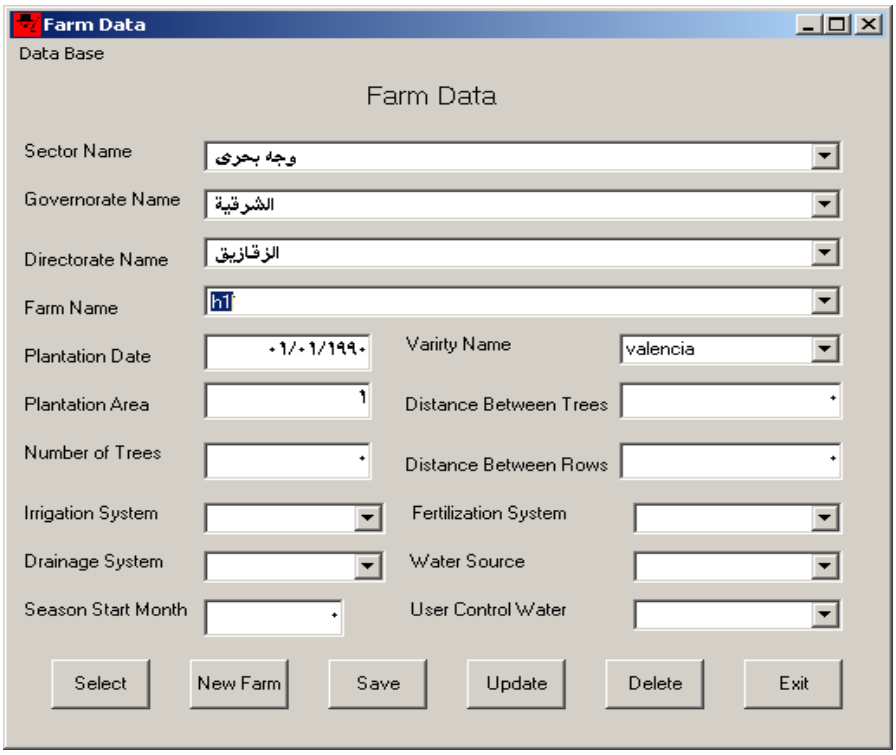
```
diag_main :-
        diag_start.

treat_main:-
        diag_start.
```

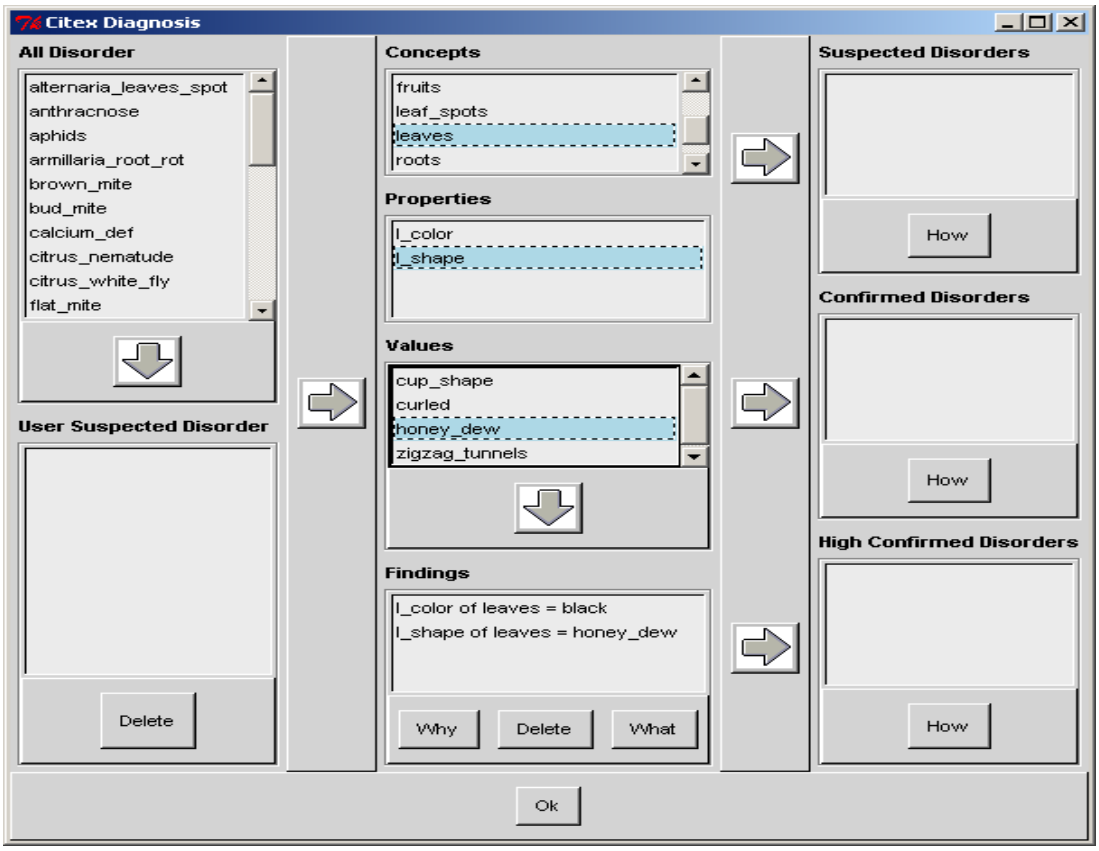
9. Test cases

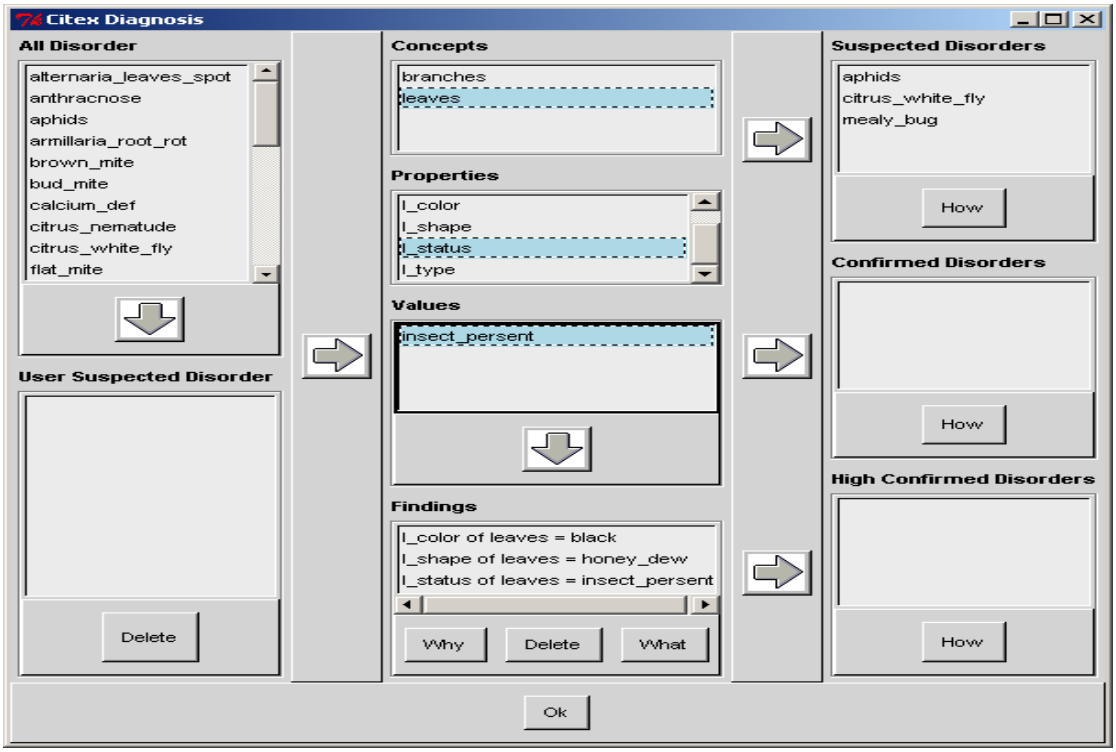
Diagnosis Test Case

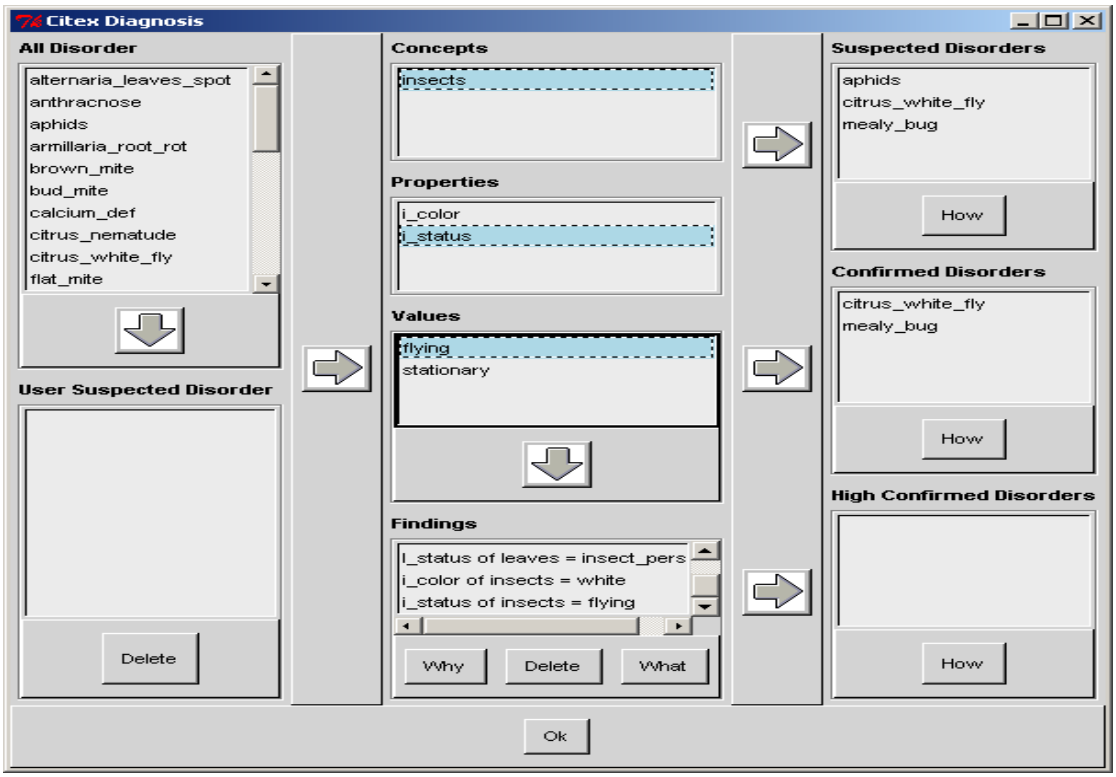
Case 1

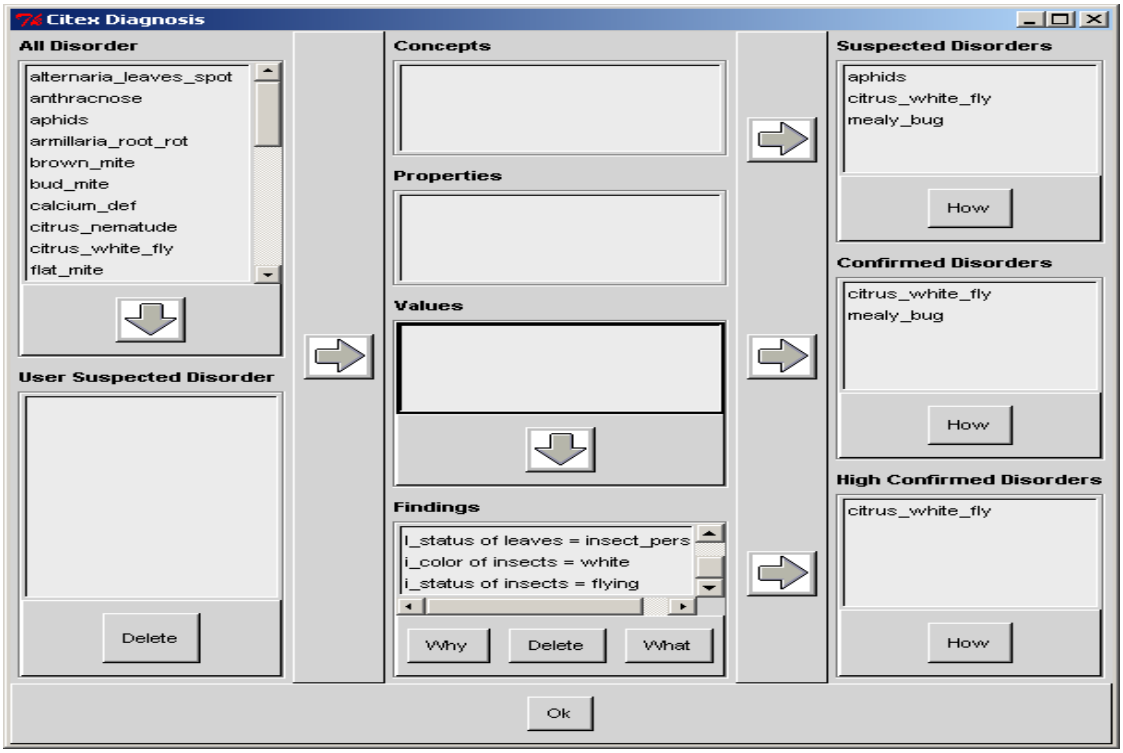
The Current Date: 1-7-01 is replaced by
Select user from database




Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
flat_mite
User Suspected Disorder
Delete
Concepts
fruits
leaf_spots
leaves
roots
Properties
l_color
l_shape
Values
cup_shape
curled
honey_dew
zigzag_tunnels
Findings
l_color of leaves = black
l_shape of leaves = honey_dew
Why
Delete
What
Suspected Disorders
How
Confirmed Disorders
How
High Confirmed Disorders
How
Ok


Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
flat_mite
User Suspected Disorder
Delete
Concepts
branches
leaves
Properties
l_color
l_shape
l_status
l_type
Values
insect_persent
Findings
l_color of leaves = black
l_shape of leaves = honey_dew
l_status of leaves = insect_persent
Why
Delete
What
Suspected Disorders
aphids
citrus_white_fly
mealy_bug
How
Confirmed Disorders
How
High Confirmed Disorders
How
Ok


Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
flat_mite
User Suspected Disorder
Delete
Concepts
insects
Properties
i_color
i_status
Values
flying
stationary
Findings
I_status of leaves = insect_pers
i_color of insects = white
i_status of insects = flying
Why
Delete
What
Suspected Disorders
aphids
citrus_white_fly
mealy_bug
How
Confirmed Disorders
citrus_white_fly
mealy_bug
How
High Confirmed Disorders
How
Ok


Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
flat_mite
User Suspected Disorder
Delete
Concepts
Properties
Values
Findings
I_status of leaves = insect_pers
i_color of insects = white
i_status of insects = flying
Why
Delete
What
Suspected Disorders
aphids
citrus_white_fly
mealy_bug
How
Confirmed Disorders
citrus_white_fly
mealy_bug
How
High Confirmed Disorders
citrus_white_fly
How
Ok

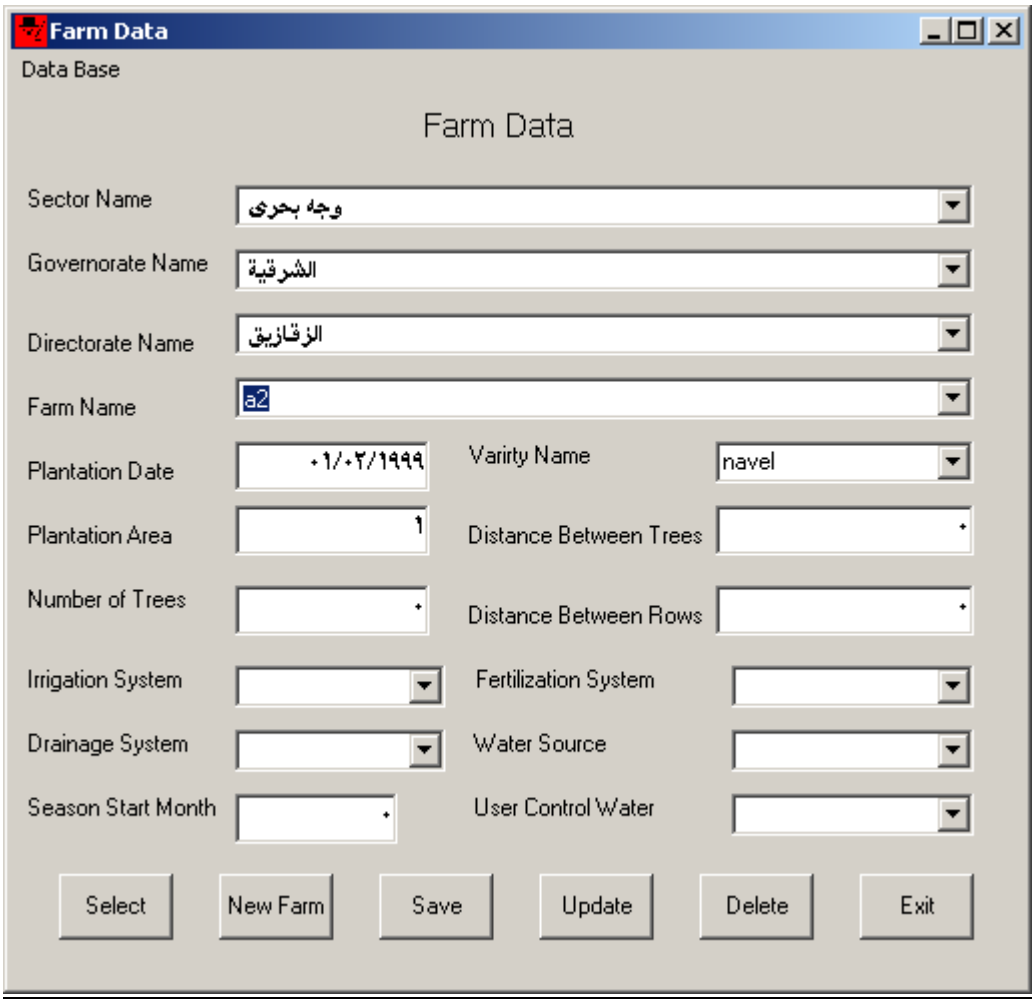
Case 2

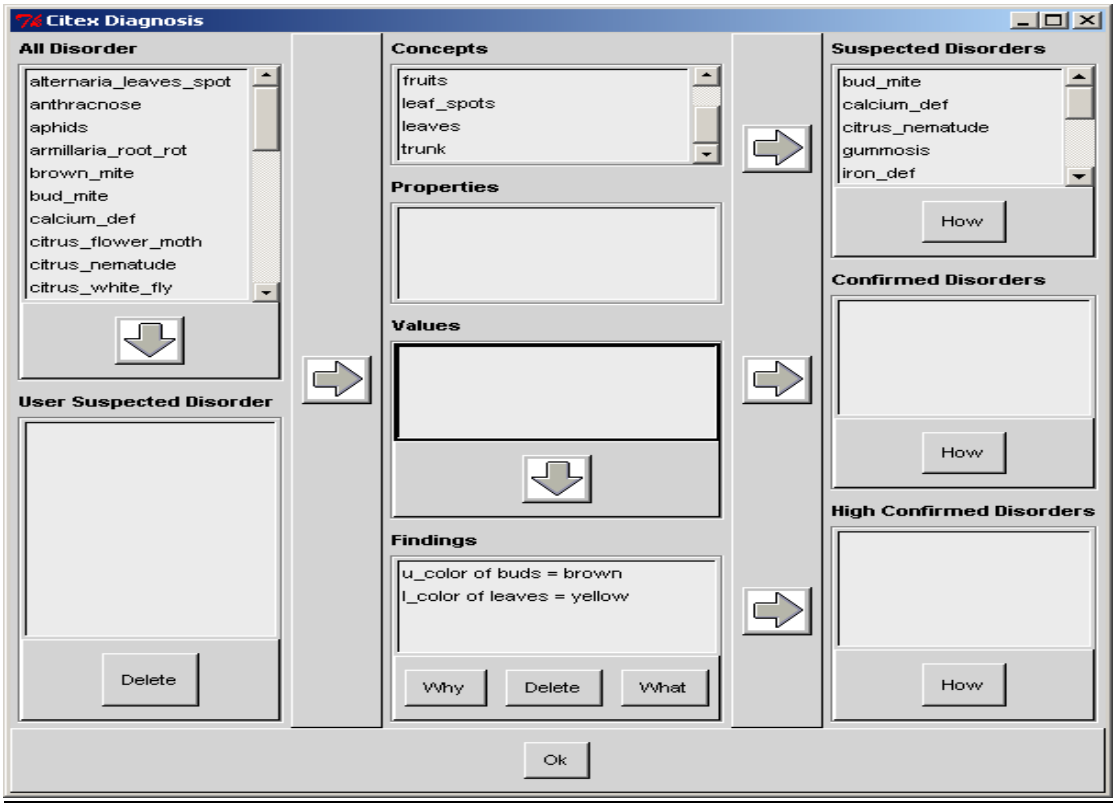
The following observation is added

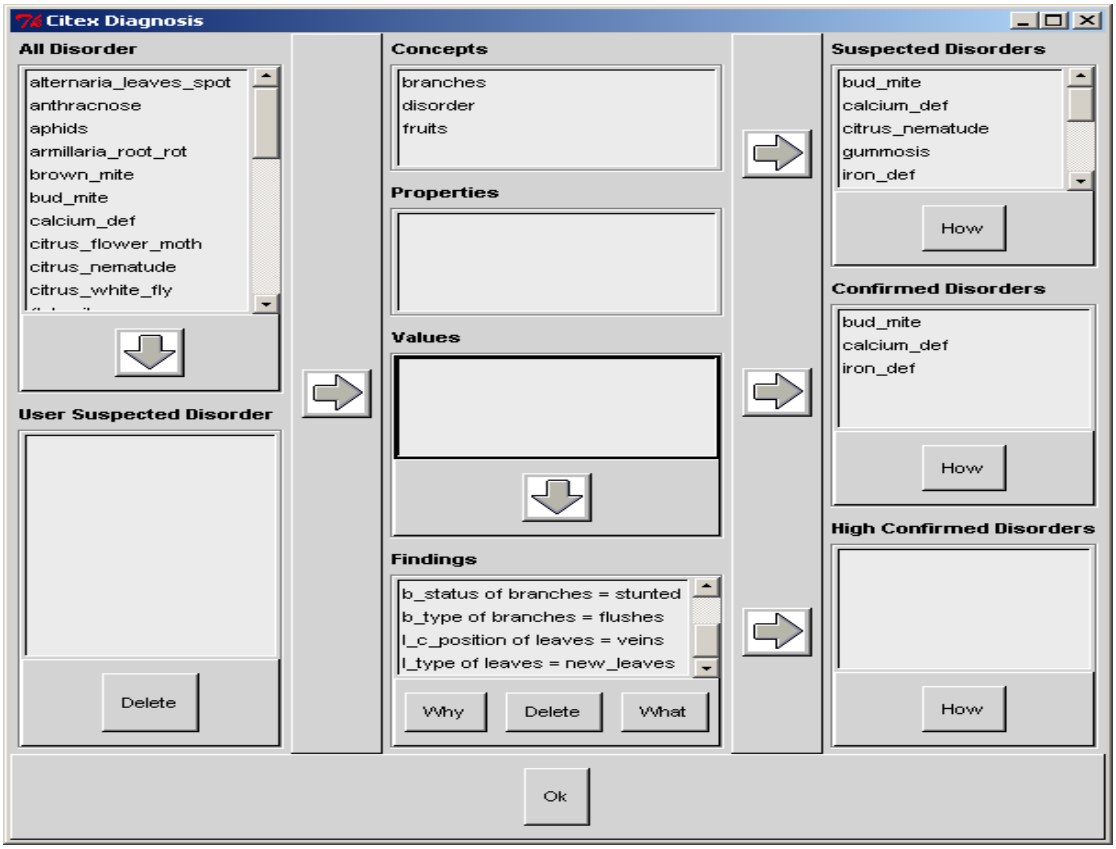
- Fruits status: reduced
- Fruits shape: small
- Buds status: abnormal

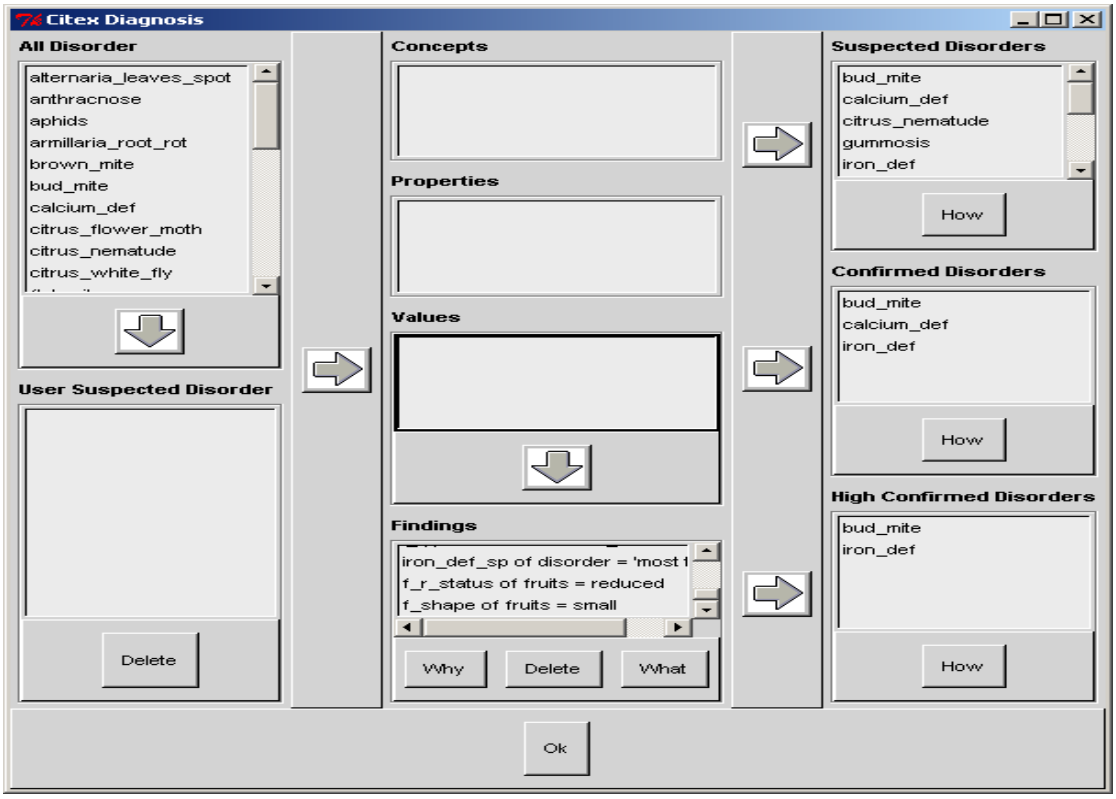
The conclusion become

- Disorders confirmed likely
  Calcium def
- Disorders confirmed most likely
  Bud mite, iron def



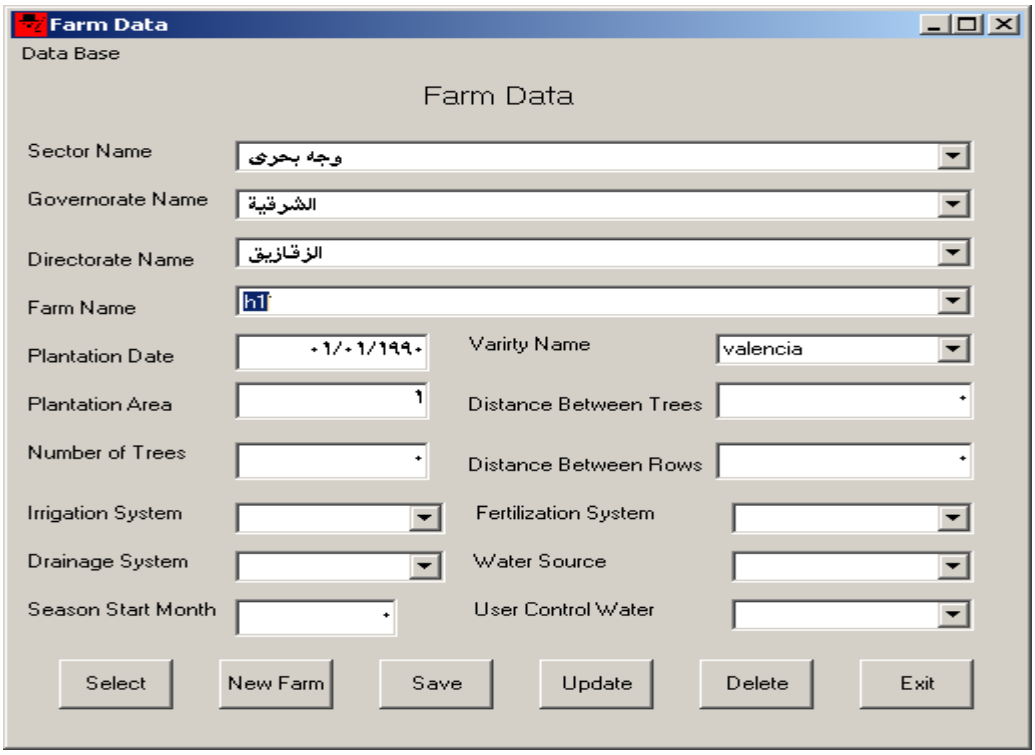

Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_flower_moth
citrus_nematude
citrus_white_fly
User Suspected Disorder
Delete
Concepts
fruits
leaf_spots
leaves
trunk
Properties
Values
Findings
u_color of buds = brown
l_color of leaves = yellow
Why
Delete
What
Suspected Disorders
bud_mite
calcium_def
citrus_nematude
gummosis
iron_def
How
Confirmed Disorders
How
High Confirmed Disorders
How
Ok

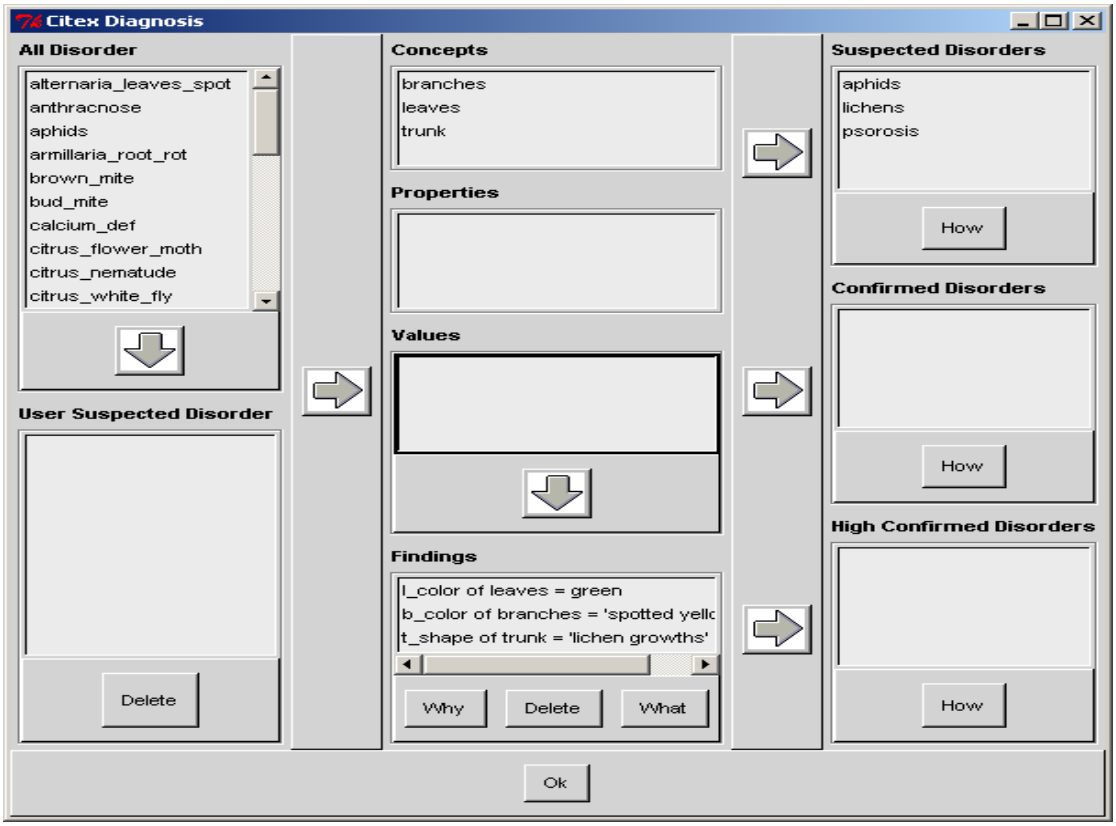

Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_flower_moth
citrus_nematude
citrus_white_fly
User Suspected Disorder
Delete
Concepts
branches
disorder
fruits
Properties
Values
Findings
b_status of branches = stunted
b_type of branches = flushes
l_c_position of leaves = veins
l_type of leaves = new_leaves
Why
Delete
What
Suspected Disorders
bud_mite
calcium_def
citrus_nematude
gummosis
iron_def
How
Confirmed Disorders
bud_mite
calcium_def
iron_def
How
High Confirmed Disorders
How
Ok

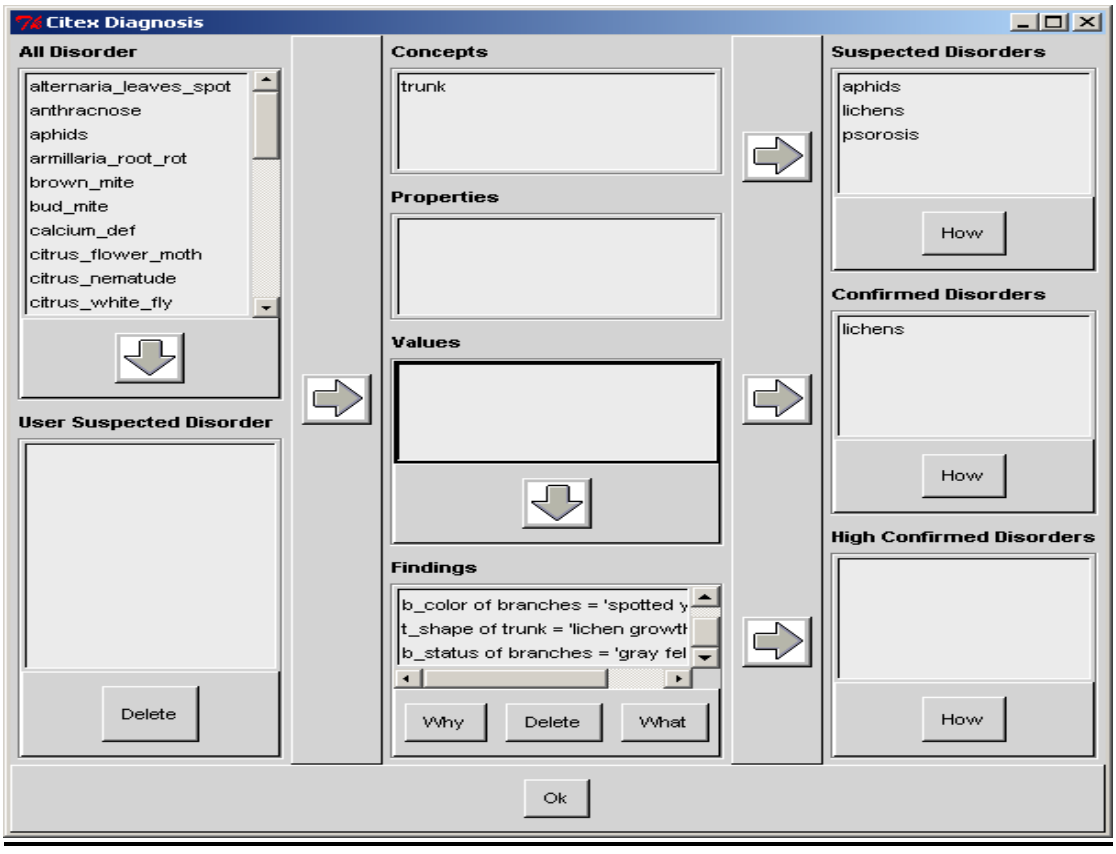


Case 3

The Plantation Date is replaced by 1-1-90 and the Current Date is replaced by 1-7-01.

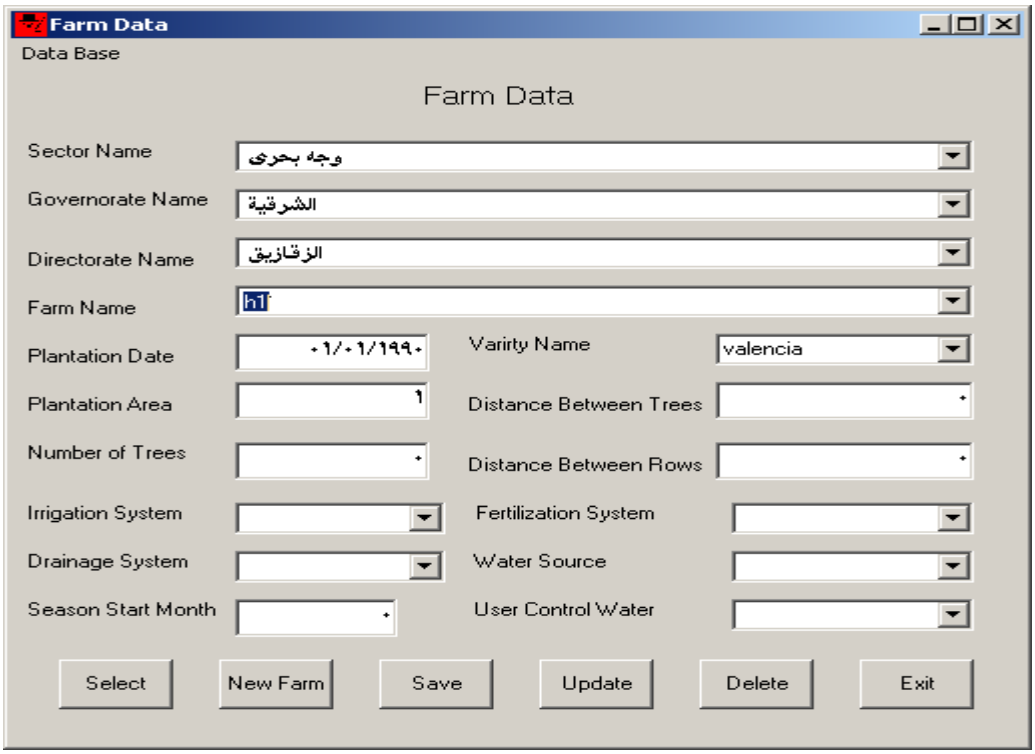


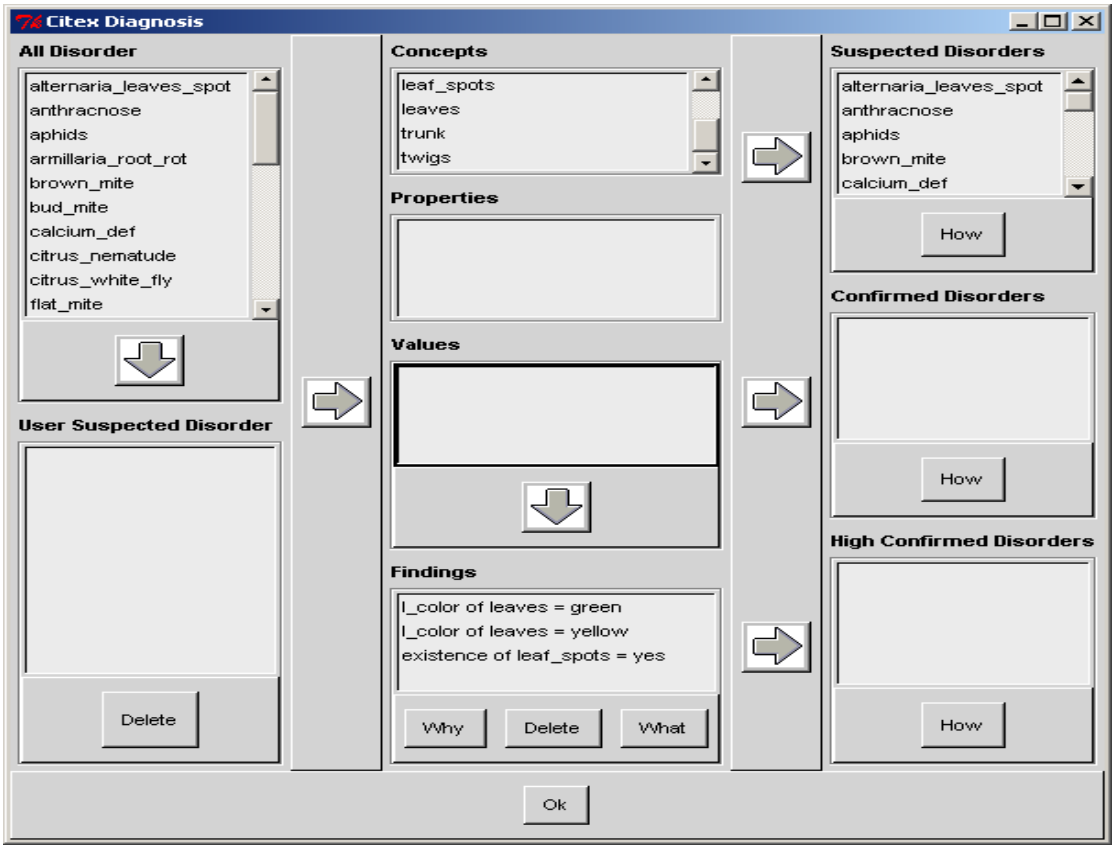

Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_flower_moth
citrus_nematude
citrus_white_fly
User Suspected Disorder
Delete
Concepts
branches
leaves
trunk
Properties
Values
Findings
l_color of leaves = green
b_color of branches = 'spotted yellc
t_shape of trunk = 'lichen growths'
Why
Delete
What
Suspected Disorders
aphids
lichens
psorosis
How
Confirmed Disorders
How
High Confirmed Disorders
How
Ok

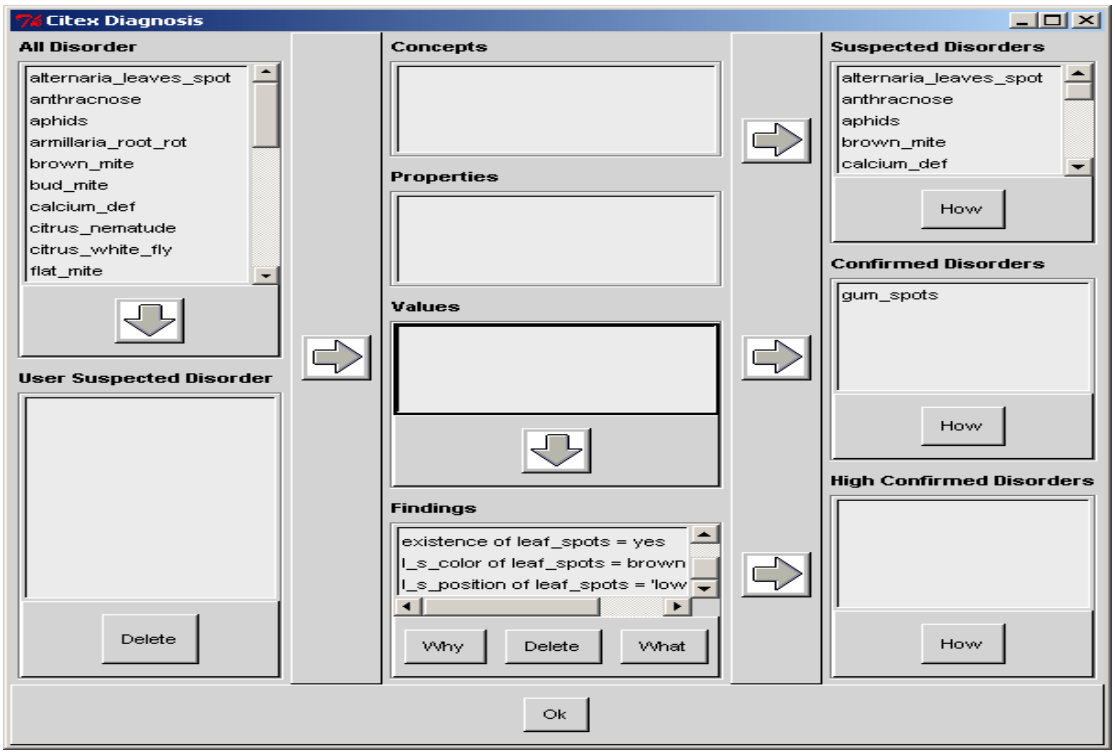

Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_flower_moth
citrus_nematude
citrus_white_fly
User Suspected Disorder
Delete
Concepts
trunk
Properties
Values
Findings
b_color of branches = 'spotted y
t_shape of trunk = 'lichen growth
b_status of branches = 'gray fel
Why
Delete
What
Suspected Disorders
aphids
lichens
psorosis
How
Confirmed Disorders
lichens
How
High Confirmed Disorders
How
Ok

Case 4

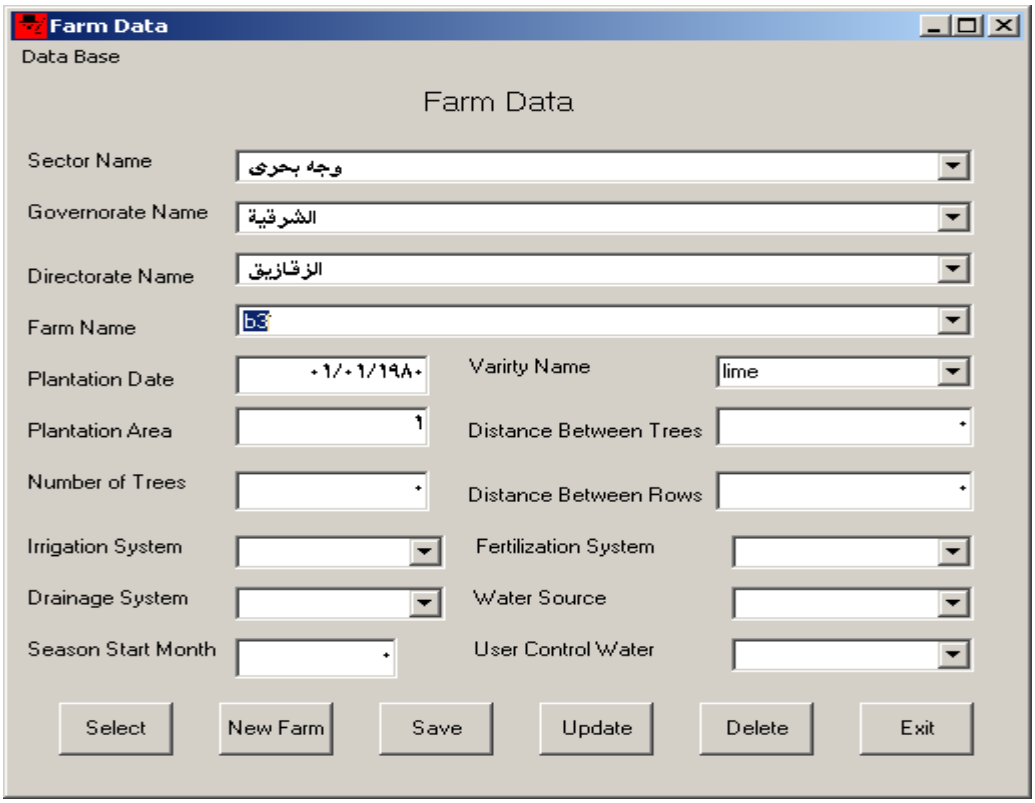
The Current Date is replaced by 1-7-01

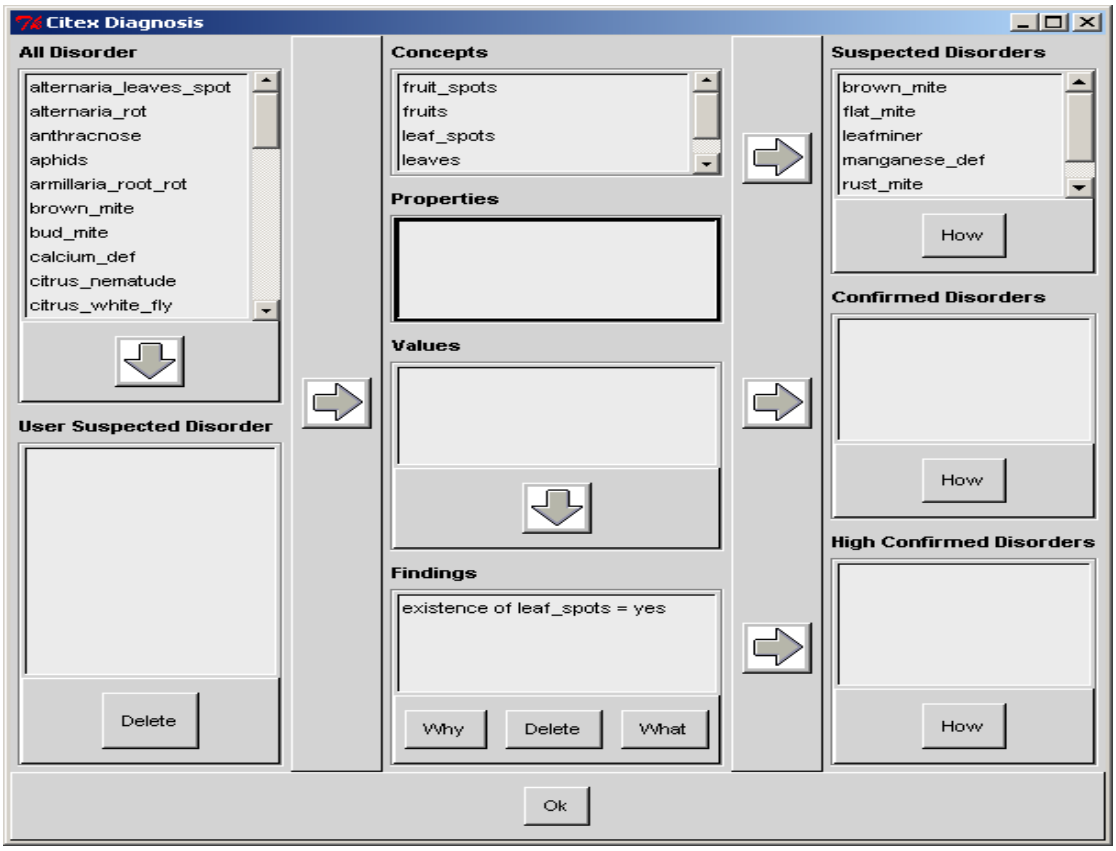


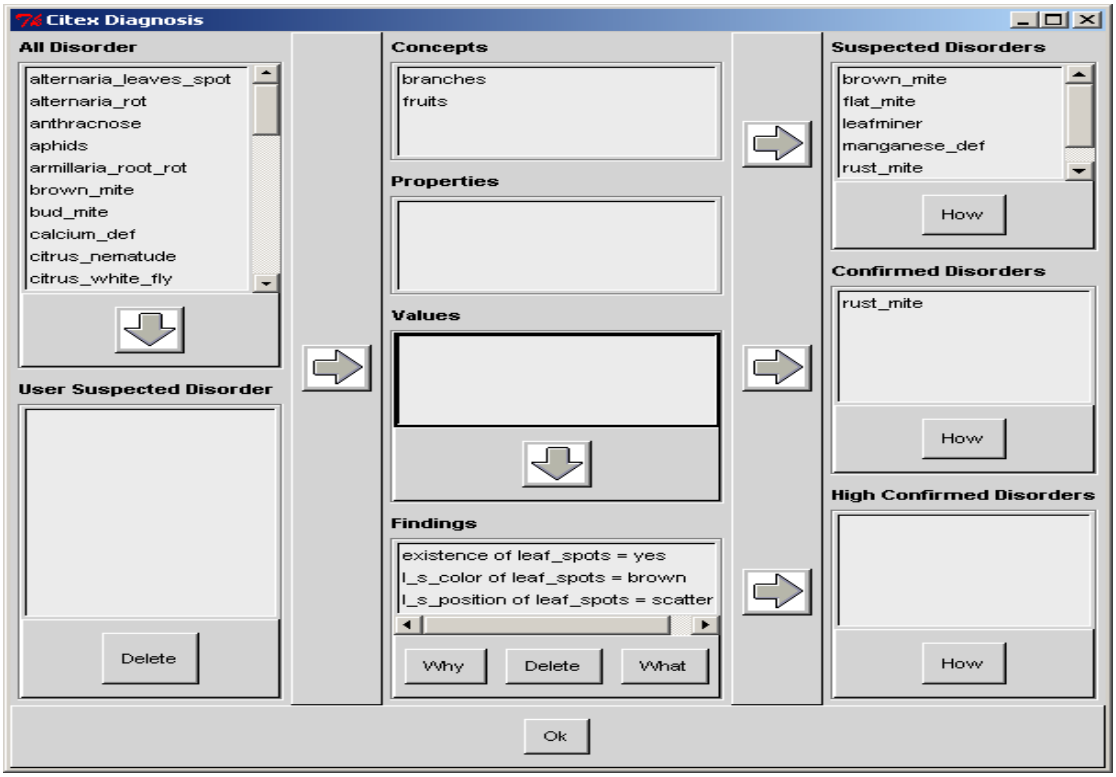


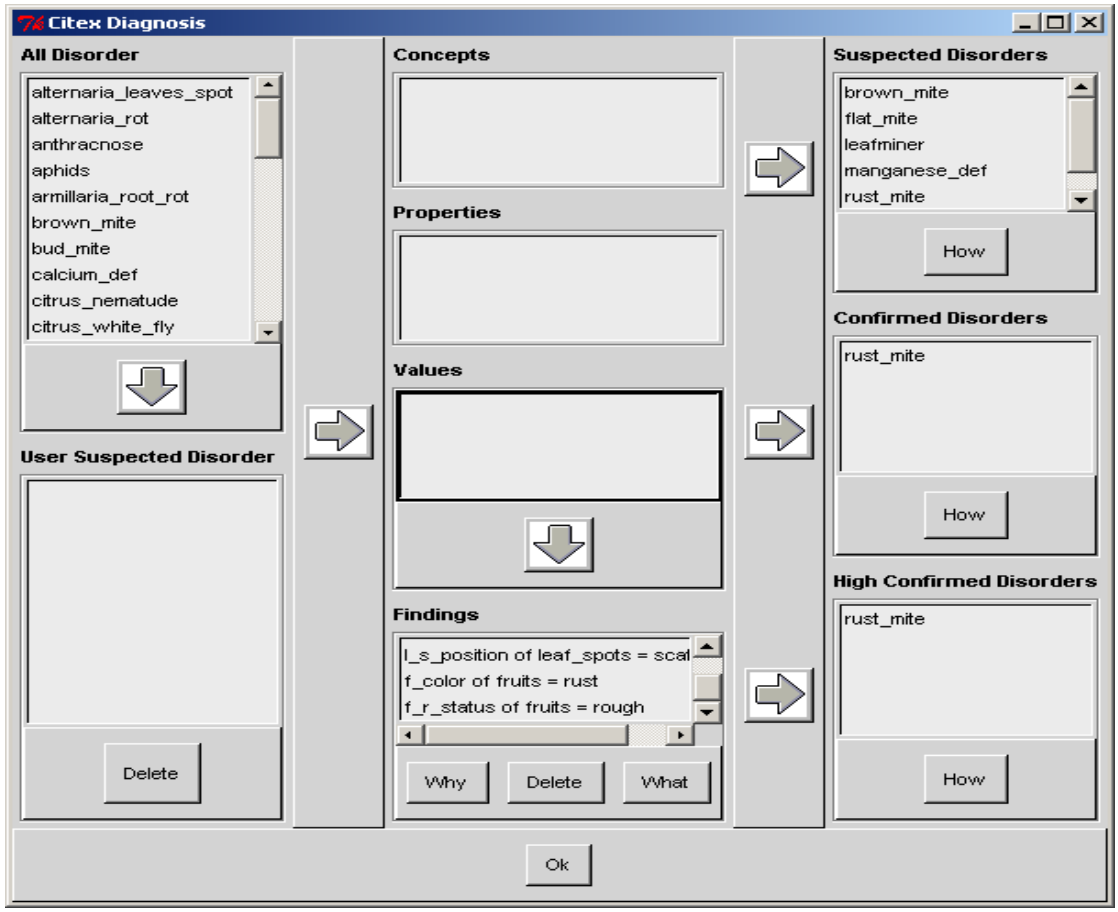

Citex Diagnosis
All Disorder
alternaria_leaves_spot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
flat_mite
User Suspected Disorder
Delete
Concepts
Properties
Values
Findings
existence of leaf_spots = yes
l_s_color of leaf_spots = brown
l_s_position of leaf_spots = 'low
Why
Delete
What
Suspected Disorders
alternaria_leaves_spot
anthracnose
aphids
brown_mite
calcium_def
How
Confirmed Disorders
gum_spots
How
High Confirmed Disorders
How
Ok

Case 5


Farm Data
Data Base
Farm Data
Sector Name
وجه بحرى
Governorate Name
الشرقية
Directorate Name
الزقازيق
Farm Name
b3
Plantation Date
٠١/٠١/١٩٨٠
Varirty Name
lime
Plantation Area
١
Distance Between Trees
Number of Trees
Distance Between Rows
Irrigation System
Fertilization System
Drainage System
Water Source
Season Start Month
User Control Water
Select
New Farm
Save
Update
Delete
Exit

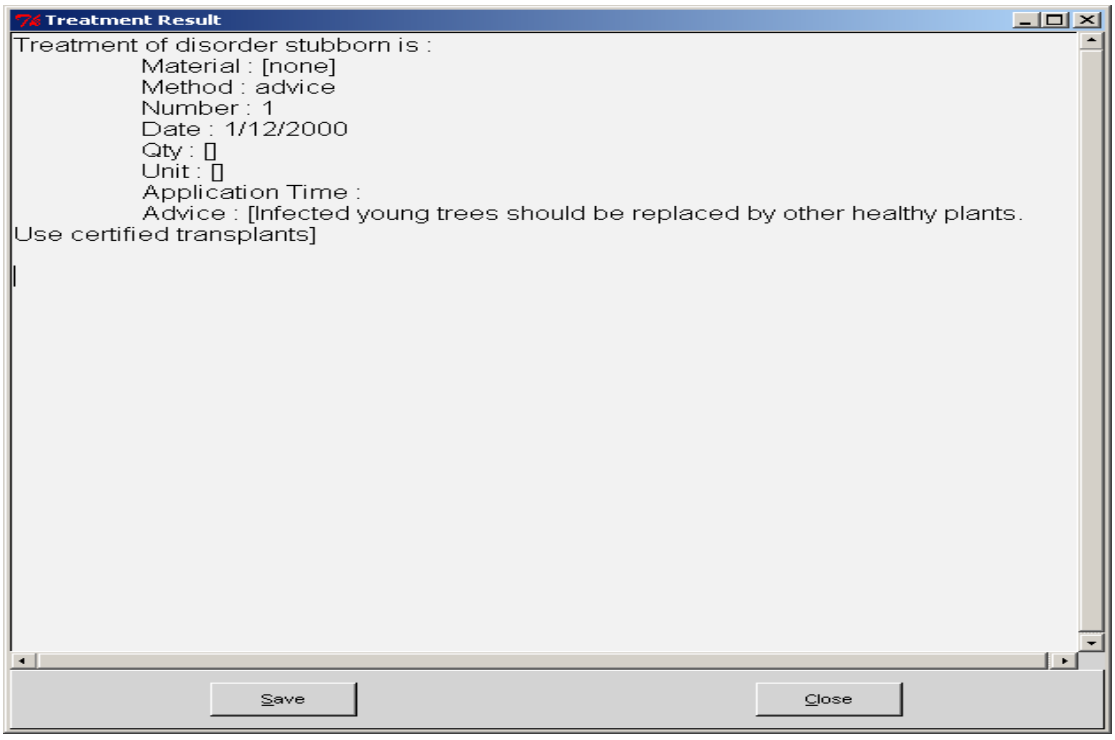

Citex Diagnosis
All Disorder
alternaria_leaves_spot
alternaria_rot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
User Suspected Disorder
Delete
Concepts
fruit_spots
fruits
leaf_spots
leaves
Properties
Values
Findings
existence of leaf_spots = yes
Why
Delete
What
Suspected Disorders
brown_mite
flat_mite
leafminer
manganese_def
rust_mite
How
Confirmed Disorders
How
High Confirmed Disorders
How
Ok


Citex Diagnosis
All Disorder
alternaria_leaves_spot
alternaria_rot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
User Suspected Disorder
Delete
Concepts
branches
fruits
Properties
Values
Findings
existence of leaf_spots = yes
l_s_color of leaf_spots = brown
l_s_position of leaf_spots = scatter
Why
Delete
What
Suspected Disorders
brown_mite
flat_mite
leafminer
manganese_def
rust_mite
How
Confirmed Disorders
rust_mite
How
High Confirmed Disorders
How
Ok


Citex Diagnosis
All Disorder
alternaria_leaves_spot
alternaria_rot
anthracnose
aphids
armillaria_root_rot
brown_mite
bud_mite
calcium_def
citrus_nematude
citrus_white_fly
User Suspected Disorder
Delete
Concepts
Properties
Values
Findings
l_s_position of leaf_spots = scal
f_color of fruits = rust
f_r_status of fruits = rough
Why
Delete
What
Suspected Disorders
brown_mite
flat_mite
leafminer
manganese_def
rust_mite
How
Confirmed Disorders
rust_mite
How
High Confirmed Disorders
rust_mite
How
Ok

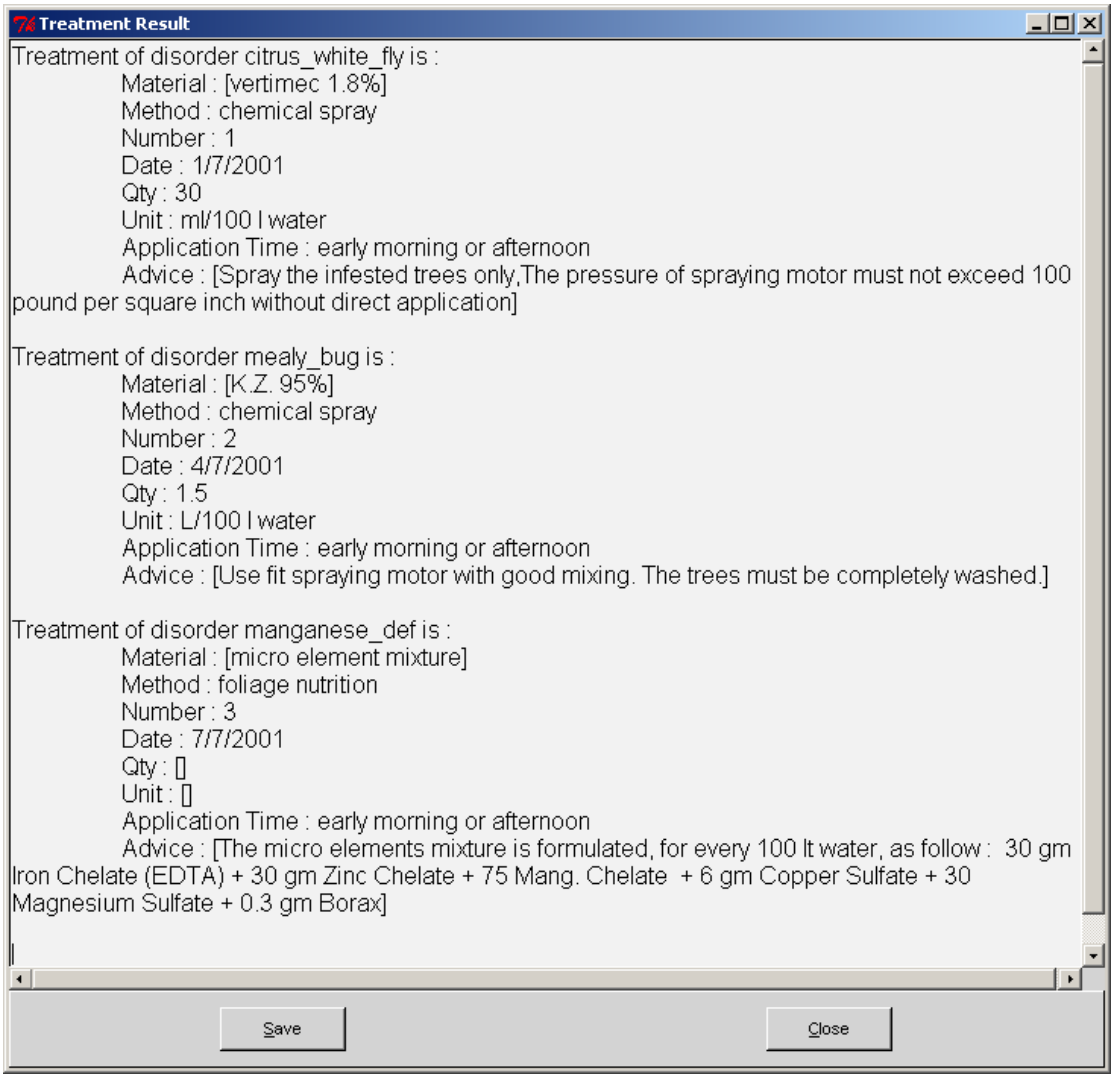
Treatment Test Case

Case 1

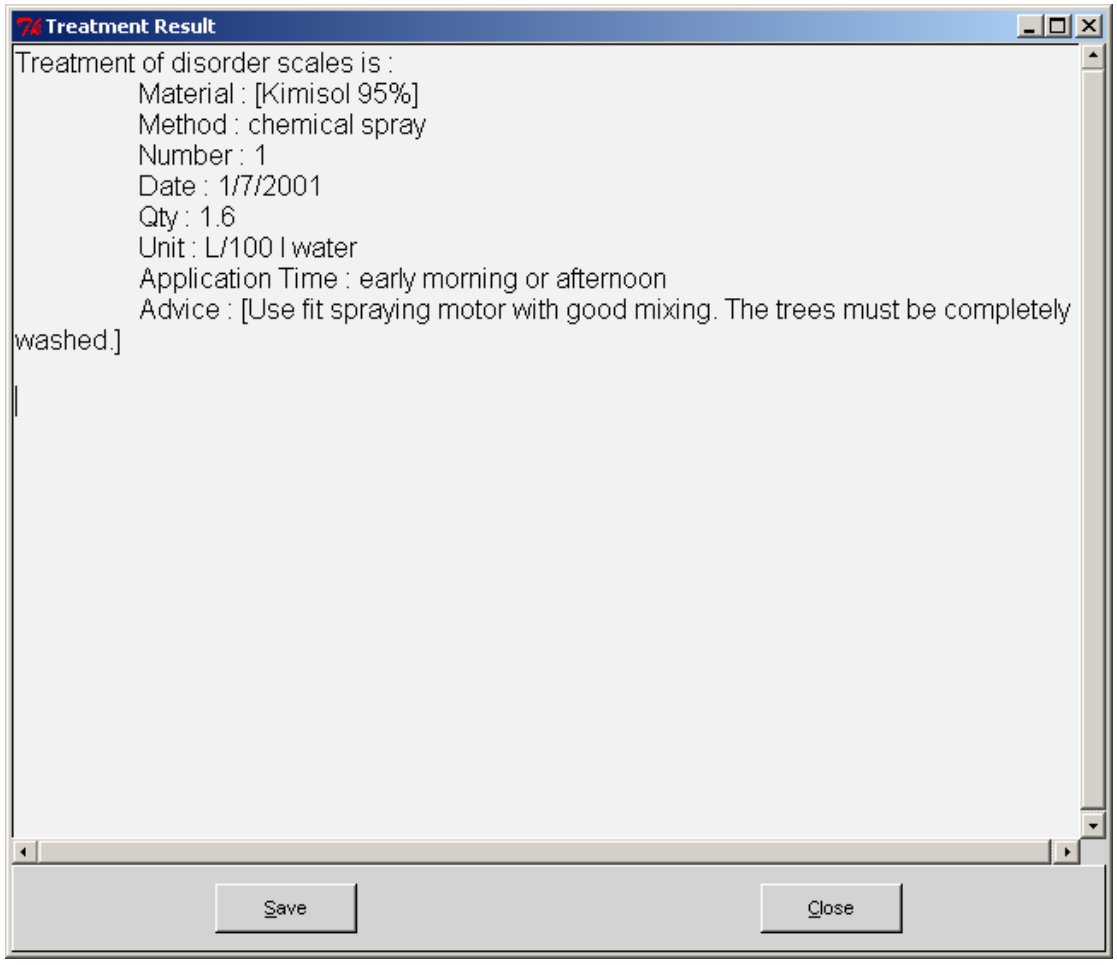


Case 2

The disorder mealy_bug is added
Select material: K.Z. 95%



Case 3


Treatment Result
Treatment of disorder scales is :
Material : [Kimisol 95%]
Method : chemical spray
Number : 1
Date : 1/7/2001
Qty : 1.6
Unit : L/100 l water
Application Time : early morning or afternoon
Advice : [Use fit spraying motor with good mixing. The trees must be completely washed.]
Save
Close

Case 4

The date is replaced by 1/4/2001