A Methodology for Building a Generic Plant Care Model

1 Introduction

The aim of this document is to present a methodology for building a generic plant care model hence the development of the appropriate tool that support a rapid plant care expert system. The generic model would serve both the developers/designers of an plant care expert system and the implementers of an expert system tool.

To do so, we have aimed to identify and capture all knowledge that is related to the plant care task, regardless of the crop variety/type, and identify concepts that vary from one crop to another. A typical plant care expert system is made out of concepts and relations, on top of which a task layer is built. By identifying concepts that are common across any plant care system, we can enhance the re-use of these concepts. We define those as static concepts. Dynamic concepts are those, which may vary from one crop to another. Though we can not accurately predict all of those, we can predict concepts to which they are related (plant, plantation type, etc) . We can consequently identify the relations that these concepts will affect and enable those to be edited to modify an irrigation model.

2 Built in/Required Concepts

2.1 concept user request

properties:

*** value**

Name	Value
Description	This concept is used by the system to give the user the chance to select the recommendation type whether based on event driven or a complete schedule.
Source of Value	User
Type	Nominal
Cardinality	Single
Legal values	العملية الزراعية التالية ، جدول كامل للعمليات

2.2 concept plant;

properties:

*** Status**

Name	value
Description	An attribute which is used to store the current plant status..
Source of Value	user
Type	nominal
cardinality	single
Legal values	Legal values should be acquired from experts

* **possible status**:

Name	value
Description	An attribute which is used to possible plant status at a specific time.
Source of Value	Derived from a relation called "**PREDICT**"
Type	nominal
cardinality	single
Legal values	Legal values should be acquired from experts

*** Age**

Name	value
Description	Plant age (will be computer from the planting date and the current date
Source of Value	derived
cardinality	single
Type	number
Legal values	0 - Max plant age

:

2.3 concept plantation;
properties:

- **area**:

Name	value
Description	Farm area in feddan
Source of Value	User
Cardinality	Single
Type	Number (real)
Legal values	+ value

- **date**:

Name	value
Description	Plantation date
Source of Value	Derived/User
Cardinality	Single
Type	Date
Legal values	

- **Farm status**

Name	value
Description	A property that describes some aspect of the current farm
Source of Value	User
Cardinality	Multiple
Type	Nominal
Legal values	**Crop dependent values**

2.4 concept operation;
properties:

method type

Name	value
Description	A property that describes the plant method plantation.
Source of Value	KE during the knowledge acquisition
Cardinality	Single
Type	Nominal
Legal values	*Mechanical, chemical, manual*

tool name:

Name	Value
Description	Tools that might be used in performing a plant care operation. The system should contain the required knowledge to determine the tool.
Source of Value	1. KE during the knowledge acquisition 2. derived if there are alternative
Cardinality	Single
Type	Nominal
Legal values	**Crop specific values**

Tool hiring rate:

Name	Value
Description	Tool hiring rate is required to compute the operation cost if it is required by the system.
Source of Value	Hiring rates might be stored in database for simplicity.
Cardinality	Single
Type	numeric
Legal values	

total time:

Name	Value
Description	Operation performing time. This property might affect either the calculation of its cost or the time of performing the next operation.
Source of Value	1. KE during the knowledge acquisition 2. Derived if it depends on the operation type
Cardinality	Single
Type	Nominal
Legal values	

labor type:

Name	Value
Description	Labor type is used to determine the number of labors per feddan to perform an operation
Source of Value	1. KE during the knowledge acquisition 2. Derived if it depends on operation type
Cardinality	Single
Type	Nominal
Legal values	**Men , boys**

number of boys per feddan:

Name	Value
Description	Number of required labor in term of working boys
Source of Value	1. KE during the knowledge acquisition 2. Derived if it depends on operation type
Cardinality	Single
Type	numeric
Legal values	**More than one**

number of men per feddan:

Name	Value
Description	Number of required labor in term of working men
Source of Value	1. KE during the knowledge acquisition 2. derived if there are alternative
Cardinality	Single
Type	numeric
Legal values	**More than one**

status

Name	Value
Description	Operation status.
Source of Value	Derived (SUGGEST relation)
Cardinality	single
Type	nominal
Legal values	not defined, suggested, not suggested
default value	Not defined

Occurrence:

Name	Value
Description	An input attribute to get information whether this operation has been done.
Source of Value	User
Cardinality	Single
Type	Nominal
Legal values	تم عملها، لم يتم عملها بعد، تم إلغائها
default value:	لم يتم عملها بعد

Importance

Name	Value
Description	Operation optional status
Source of Value	1. KE during the knowledge acquisition 2. Derived if there are alternative
Cardinality	Single
Type	Nominal
Legal values	إجبارية، اختيارية

method

Name	value
Description	Text that describe the operation method
Source of Value	Derived from a relation called "**Assign**"
Type	nominal
cardinality	single

operation number

Name	value
Description	This property is used to identify the operation number. And is used to sort the operations.
Source of Value	Static – to be assigned by the KE
cardinality	single
Type	number

:

cost

Name	value
Description	Operation cost.
Source of Value	KE during the knowledge acquisition derived if it depended on other factors
Cardinality	Single
Type	Number (real)
Legal values	+ value

actual cost

Name	value
Description	Operation cost.
Source of Value	User
Cardinality	Single
Type	Number (real)
Legal values	+ value

name

Name	value
Description	Operation name
Source of Value	Derived (Apply)
Cardinality	multiple
Type	nominal
Legal values	(Listing of possible crop operations)

Time difference

Name	value
Description	The number of days that should separate two successive operations
Source of Value	KE
Cardinality	Single
Type	number
Legal values	

Application date

Name	value
Description	The operation application date
Source of Value	User at run time/control
Cardinality	Single
Type	date

2.5 concept operation cost;
properties:

value

Name	value
Description	Property to get from the user whether he/she need to compute the total operations costs
Source of Value	User
Cardinality	single
Type	Nominal
Legal values	مطلوبة ,غير مطلوبة

2.6 concept seedling;
properties:
sub-type-of: plant;

type

Name	value
Description	
Source of Value	User or database
Cardinality	Single
Type	Nominal
Legal values	Crop dependent

2.7 concept session date;
properties:
current month

Name	Value
Description	
Source of Value	System
Cardinality	Single
Type	Nominal

value

Name	Value
Description	
Source of Value	System
Cardinality	Single
Type	date

2.8 concept Irrigation;
properties:
type

Name	Value
Description	Irrigation type used in the farm
Source of Value	database
Cardinality	Single
Type	numeric
Legal values	الري بالغمر، الري بالتنقيط

2.9 concept Planting;
properties:
type

Name	Value
Description	Planting type
Source of Value	database
Cardinality	Single
Type	Nominal
Legal values	زراعة على مصاطب، زراعة على خطوط

2.10 concept event;
properties:

occurrence

Name	Value
Description	
Source of Value	User
Cardinality	Single
Type	Nominal
Legal values	حدث ، لم يحدث
default value	لم يحدث

value

Name	Value
Description	The current event
Source of Value	user
Cardinality	multiple
Type	nominal
Legal values	**Crop dependent**

2.11 concept soil;
properties:

type

Name	Value
Description	
Source of Value	database
Cardinality	Single
Type	nominal
Legal values	خفيفة، ثقيلة، متوسطة

2.12 concept previous crop;
properties:
fertilizer

Name	Value
Description	Information regarding the previous crop fertilization
Source of Value	database
Cardinality	Single
Type	nominal
Legal values	لم يسمد بكمية سماد عضوي جيدة، تم تسميده بكمية سماد عضوي جيدة

2.13 Operations concept instances

In this section the knowledge engineer should specify all instances of the plant care operations that should be done during the crop lifetime. The following template might be used to fill these information:-

Instance name
sub-type-of: operation;
[list of all properties and their corresponding associated values that are acquired from the expert about the operation**]**

2.14 Event instances
In this section the KE state the events that have an effects on the occurrence of the operations. The following form will be used:

Instance name
sub-type-of: event;

3. Domain Models

The plant care system include 4 domain models:-

1. **Predict:** This model is used to derive the plant possible current stages (by generate plant status inference step- see figure 2)
2. **Suggest:** this model is used to infer the suggested operation. It contain the two types of conditions:
    a. **Sequence condition**, which enforce the sequence on the operations such that keep the operation done in the proper way. Example of the sequence conditions is

<previous op name. Occurrence> = done &
<current op name. Occurrence> = not done yet

another example for the sequence condition and is used to permits a time difference between two successive operation is as followed

<session date. Value> >= <previous operation-name> . application date +
< previous operation-name>. time difference>

b. **Environment conditions:** any environment condition such that: -
    1. Time or plant age
    2. Plantation or farm condition
    3. Current crop state
    4. Event occurrence

3. **Assign :** This model is used to fill the operation properties that should be displayed to the user of the system such that:
    - Operation method
    - Operation tool
    - Operation material
    - Material quantities
    - Number of labor per feddan
    - Operation cost
4. **Recommend**: this domain model will be used by the determine inference step (see figure 2) to generate all possible operations that are applicable to a specific farm conditions. Note that this inference is only used in case a complete operation schedule is required by the system user.

The model schema is as follows
< environment conditions> -- > operation . name. = <op-instance>

3.1 Operation state diagram

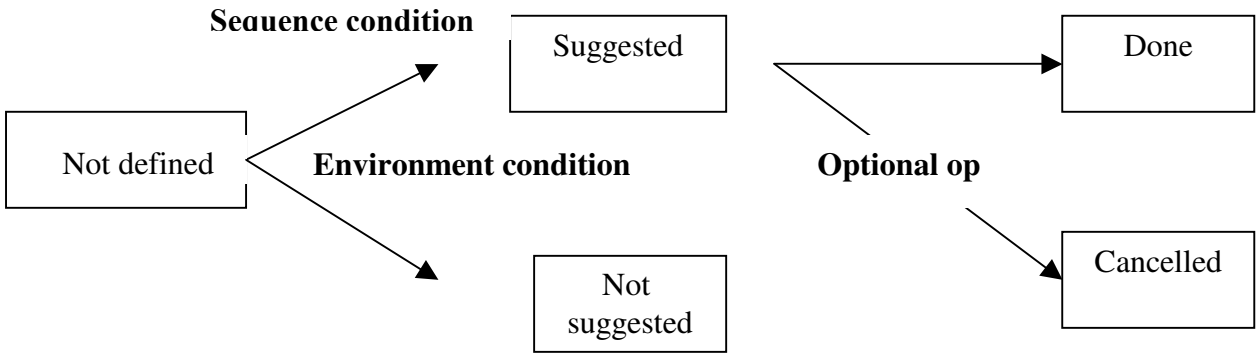
We can distinguish 2 types of operations :

a. **Obligatory operation** its definition is " an operation if its condition (either environment condition or sequence condition) is met, it should be done, i.e., No way for canceling.

b. **Optional operation**: its definition is "an operation if its condition (either environment condition or sequence condition) is met, it may be done canceled by user.

3.2 Operation states

We have the following operation states diagram



4. Inference Knowledge

4.1 Inference Structure

Inference structure is shown in figure 2

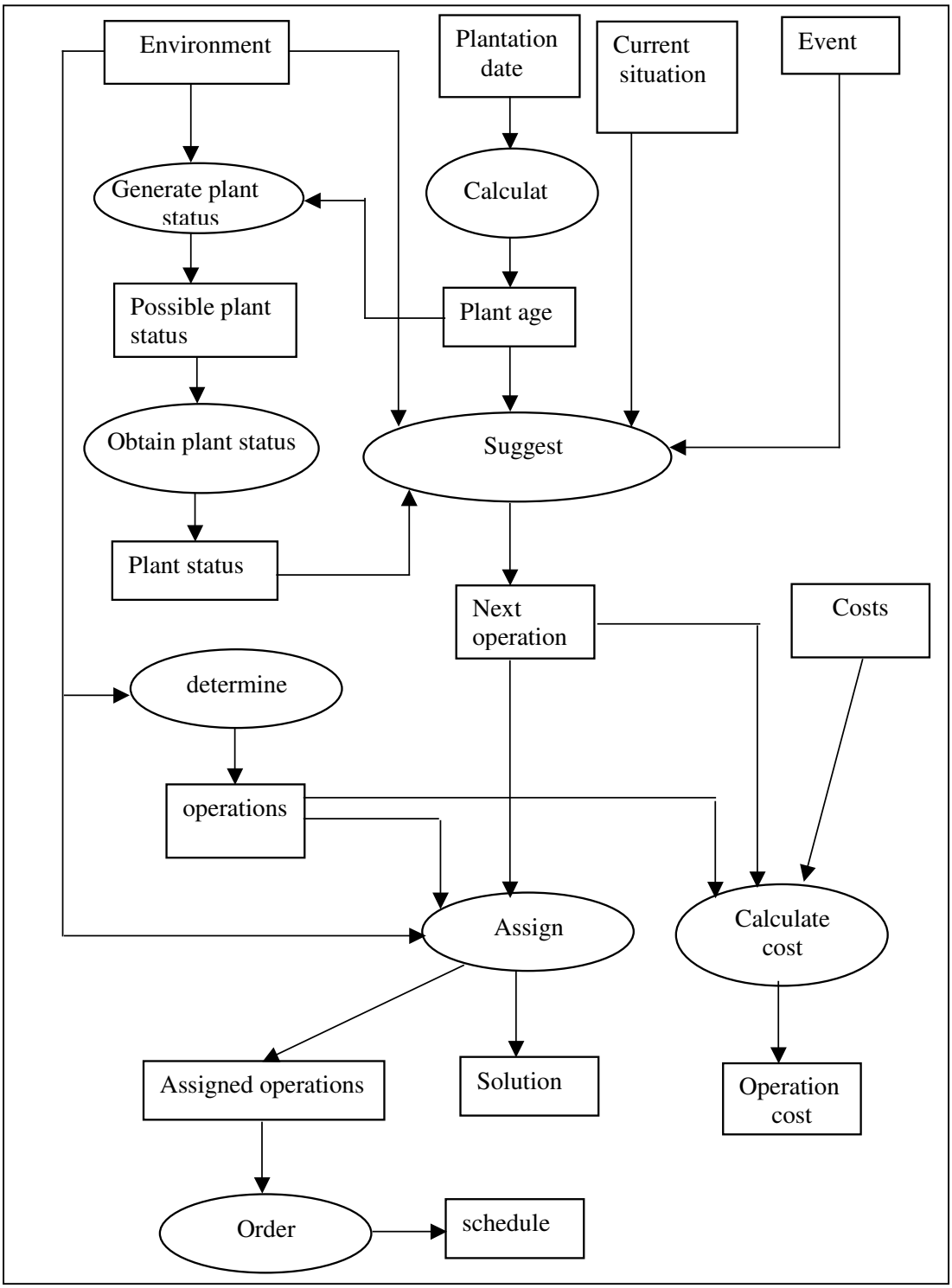


Figure 2: inference structure

4.2 Inference specification

inference: calculate
operation-type: calculate the plant age.
input-roles: plantation date.
output-roles: plant age.
static-roles: there is no static roles
spec: plant age is calculated in days by subtracting the plantation date from the session date.

inference: generate plant status
operation-type: generates the possible plant status.
input-roles: environment, plant age.
output-roles: possible plant status.
static-roles: PREDICT ∈ prediction–model.
spec: the possible plant status are generated by applying "PREDICT " relation.

inference: obtain plant status
operation-type: transfer task.
input-roles: possible plant status.
output-roles: plant status.
static-roles: there is no static roles.
spec: obtain the current plant status from the user.

inference: suggest
operation-type: suggest the next agricultural operation.
input-roles: environment, plant age, event, plant status, current situation.
output-roles: next operation.
static-roles: SUGGEST ∈ suggestion–model.
spec: the next agricultural operation are to be suggested by applying " SUGGEST " relation.

inference: determine
operation-type: determine the agricultural operations schedule.
input-roles: environment.
output-roles: operations.
static-roles: APPLY ∈ application model.
spec: the agricultural operations schedule is determined by applying " APPLY" relation.

inference: assign
operation-type: assign parameters to the suggested operation.
input-roles: next operation, environment, operations.
output-roles: solution, assigned operations.
static-roles: ASSIGN ∈ assignment–model.
spec: assign the method to the suggested operation or to the scheduled operations

by applying " ASSIGN " relation.

inference: order
operation-type: order the operations ascending according to
operation: operation number .
input-roles: assigned operations
output-roles:. schedule
static-roles: there is no static role
spec: get operation: name(L), /* L is a list */
for each element(E) in L get E: operation number,
sort ascending L according to operation number of each element,
the sorted list is L',
DISPLAY(L')

inference: calculate cost
operation-type: calculate cost of the suggested operation.
input-roles: next operation, costs.
output-roles: operation cost.
static-roles: there is no static role
spec:
IF the user request is (next operation)
THEN
Begin
IF the method of the next operation is mechanical
THEN get the tool name (T) of the next operation
get the tool hiring rate (THR) of (T)
get the total time/Feddan (TT) of applying the next operation
get the area (A) of the plantation
cost = THR*TT*A
IF the method of the next operation is manual
THEN get the number of men/Feddan (M) of applying the next operation
get the labor wage (W)
get the area (A) of the plantation
cost = M*W*A
End
ELSE
Begin
get operation: name (N) % N is a list
While N <> { }
Begin
N = [E | Tail]
IF the method of 'E' is mechanical
THEN get the tool name (T) of 'E'
get the hiring rate (HR) of (T)
get the total time/Feddan (TT) of applying 'E'
get the area (A) of the plantation
Cost = HR*TT*A
Insert Cost in E: cost

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

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

5. Task Knowledge

task plant care*;*
task-definition:
goal: suggest the next agricultural operation,
presents a schedule of the agricultural operations;
input: Environment: {soil: type, seedling : type, previous crop : fertilizer,
Irrigation: type, Planting: type };
Environment: { plant: status};
plantation date: {plantation: date};
current situation: {operation: occurrence};
event: {event: value};
cost: {the cost of equipments, labor , … etc from database}
output: solution: { suggested operation, importance, method},
operation cost: {cost of the suggested operation},
schedule: {the schedule of the agricultural operation during the season},
task-body:
type: composite
subtasks: calculate, generate plant status, obtain plant status, suggest, assign, determine, order,

additional-roles:
Possible plant status: {plant: possible status }
Plant status {plant: status}
plant age: {plant: age}
Next operation {operation: status = suggested}
Operations {operation: name}
Assigned operations
control-structure:

{
Case user request: value
= العملية الزراعية التالية

{

Calculate plant age (plantation date→ plant age)
Generate plant status (environment → Possible plant status)
Obtain plant status (plant status)
Recommend (environment → recommended operations)
Suggest (recommended operation,
environment, plant age,
current situation,
plant status,
event → next operation)

IF operation cost: value = مطلوبة
THEN calculate cost(O, C: costs ⟶ OC: operation cost)
calculate cost (next operation, costs /* from user *./ → operation cost)
Assign (next operation → solution)

}

جدول كامل للعمليات=

determine(E: environment → O: operations),
Assign(E, O → AP: assigned operations),
order(AP → S: schedule),

IF operation cost: value = مطلوبة
THEN calculate cost(O, C: costs ⟶ OC: operation cost)
PRESENT(S, OC)
ELSE PRESENT(S)

}

}