Ministry of Agriculture & Land Reclamation
Agricultural Research Center
Central Lab for Agricultural Expert Systems

Detailed Design for Plant Care Task Template

TRICLAES/227/2001.8

By

Eng.Mohammed El Helly
Eng. Mohammed Yehia

August, 2001

1- Introduction

The objective of this report is to describe the template design of the interface, which is used in the plant care subsystem. Also the event handlers associated with each interface components are described in more details. Mainly the plant care subsystem is divided into two dialogues, the first dialogue is the database dialogue and the second one is the plant care dialogue. This report consists of seven sections the interface design of database in section two and interface design of the main dialogue for the plant care in section three, section four describes the proposed operation dialogue, section five describes the operation detail dialogue, section six shows the previous operation dialogue and finally the conclusion in section seven.

2- Interface Design of Database

This Section describe the design of the database dialogue and the event which are handled in this dialogue and the algorithms associated with each event

2-1 Dialogue Design

Figure 1

The above Figure shows the design of the database dialogue, which consists of six combo boxes and four push buttons.

2-2 Event Used in database dialogue:

The following are the events which is used in the database dialogue:

- On Initiate Dialogue.
- On Change Selection of Farm Combo Box.
- On Click New Farm Button.
- On Click Add/Update Button
- On Delete Button.
- On Exit Button.

2-3 Algorithms Associated With the Event:

The following algorithms are the event handler associated with the above events.

On Initiate Dialogue

```
Begin
        SeedsStatusLegal=  GetFromKB(“seeds” , “status”);
        AddComboBoxItem(SeedsStatusCombo, SeedsStatusLegal)
        SoiltypeLegal = GetFromKB(“soil” , “type”);
        AddComboBoxItem(SoiltypeCombo, SoiltypeLegal)
        PeviousCropFertilizerLegal = GetFromKB(“previous_crop” , “fertilizer”);
        AddComboBoxItem(PreviousCropFertilizerCombo, PeviousCropFertilizerLegal)
        PlantationCropTypeLegal = GetFromKB(“plantation” , “crop_type”);
        AddComboBoxItem(PlantationCropTypeCombo, PlantationCropTypeLegal)
        PlantationTypeLegal = GetFromKB(“plantation” , “type”);
        AddComboBoxItem(PlantationTypeCombo, PlantationTypeLegal)
        FarmNames = GetFromPlantCareDB(Farms)
        AddComboBoxItem(FarmNamesCombo, FarmNames);
        AddComboBoxItem(PlantationDateCombo, Now);
End
```

On Change Selection of Farm Combo Box

```
Begin
        FarmName = GetCurrentSelectionText(FarmNamesCombo);
        FarmObject = LookupFromPlantCareDB(FarmName)
        SetCuurentSelectionText( SeedsStatusCombo, FarmObject->SeedsStatus)
        SetCuurentSelectionText(SoiltypeCombo, FarmObject->SoilType)
        SetCuurentSelectionText(PreviousCropFertilizerCombo, FarmObject->PreviousFertilizer)
        SetCuurentSelectionText(PlantationCropTypeCombo, FarmObject->PlantationCropType)
        SetCuurentSelectionText(PlantationTypeCombo, FarmObject->PlantationType)
        SetCuurentSelectionText(PlantationDateCombo, FarmObject->PlantationDate)
End
```

On Click New Farm Button

```
Begin
        SetCurrentSelectionText(FarmNamesCombo, "");
        SetCuurentSelectionText(SeedsStatusCombo, "")
        SetCuurentSelectionText(SoiltypeCombo, "")
        SetCuurentSelectionText(PreviousCropFertilizerCombo,"")
        SetCuurentSelectionText(PlantationCropTypeCombo,"")
        SetCuurentSelectionText(PlantationTypeCombo, "")
        SetCuurentSelectionText(PlantationDateCombo, "")
End
```

On Click Add/Update Button

```
Begin
        FarmName = GetCurrentSelectionText(FarmNamesCombo);
        FarmObj = CreateNewObject(FarmName);
        FarmObj->SeedStatus =GetCuurentSelectionText(SeedsStatusCombo);
        FarmObj->SoilType = GetCuurentSelectionText(SoiltypeCombo);
        FarmObj->PreviousCropFertilizer = GetCuurentSelectionText(PreviousCropFertilizerCombo);
        FarmObj->PlantationCropType = GetCuurentSelectionText(PlantationCropTypeCombo);
        FarmObj->PlantationType = GetCuurentSelectionText(PlantationTypeCombo);
        FarmObj->PlantationDate = GetCuurentSelectionText(PlantationDateCombo);
        AddToPlantCareDB(FarmObj);
        SetToWM("seeds" , "status", FarmObj->SeedStatus);
        SetToWM ("soil" , "type", FarmObj->SoilType);
        SetToWM ("previous_crop" , "fertilizer", FarmObj->PreviousCropFertilizer);
        SetToWM ("plantation" , "crop_type", FarmObj->PlantationCropType);
        SetToWM ("plantation" , "type", FarmObj->PlantationType);
        SetToWM ("plantation" , "date", FarmObj->PlantationDate);
End
```

On Click Delete Button

```
Begin
        FarmName = GetCurrentSelectionText(FarmNamesCombo);
        RemoveFromPlantCareDB(FarmName);
        RemoveFromWM(“seeds” , “status”);
        RemoveFromWM (“soil” , “type”);
        RemoveFromWM (“previous_crop” , “fertilizer”);
        RemoveFromWM (“plantation” , “crop_type”);
        RemoveFromWM (“plantation” , “type”);
        RemoveFromWM (“plantation” , “date”);
        CallEvent(OnNewFarmButton);
End
```

On Click Exit Button

```
Begin
        If(FarmData is not Empty)
                CloseDialoge(DBDialoge);
End
```

3 Interface Design of Plant Care

This Section describe the design of the main dialogue of the plant care subsystem and the event which are handled in this dialogue and the algorithms associated with each event

3-1 Dialogue Design

Figure 2

The above Figure shows the design of the plant care dialogue, which consists of four list boxes, five radio buttons, and four push buttons.

3-2 Event Used in Plant Care Dialogue:

The following are the events which is used in the database dialogue:

- On Initiate Dialogue.
- On Click Suggested Operation Button.
- On Double Click of Plant Status List Box.
- On Change Selection of Proposed Operation List Box.

- On Change Selection of Event List Box.
- On Select Operation Done Radio Button.
- On Select Operation Not Done Radio Button.
- On Select Operation Cancel Radio Button.
- On Select Event Done Radio Button.
- On Select Event Not Done Radio Button.
- On Click Previous Done Operation Button
- On Exit Button.

3-3 Algorithms Associated With the Event:

The following algorithms are the event handler associated with the above events.

On Initiate Dialogue

```
Begin
        DOTransfereTask(DB Dialogue);
        CallInference(Calculate Growth Stage);
        CallInference(Generate Plant Status);
        ClearListBox(PlantStatusListBox)
        ClearListBox(PodStatusListBox)
        ClearListBox(EventListBox)
        PlantAge = GetValue("plant"," age");
        If(PlantAge  >10 AND PlantAge <50)
                DiablePodListBox();
        Else{
                EnablePodListBox();
                PodStatusList = GetValue("pods"," possible_status");
                FillListBox(PodStatusListBox, PodStatusList)
        }
        PlantStatusList = GetValue("plant"," possible_status");
        FillListBox(PlantStatusListBox, PlantStatusList )
        PlanationType = GetValue("plantation"," type");
        If (PlanationType == "Open Field")
                DiableEventListBox();
        Else{
                EnableEventListBox();
                EventList = GetFromKB("event","value");
                FillListBox(EventListBox, EventList);
        }
End
```

On Click Suggested Operation Button

```
Begin
        For(I = 0; I< GetCount(SuggestedOperationListBox);I++){
            Operation = GetListBoxItem(I, SuggestedOperationListBox)
            OpOccurance = GetValue(Operation, " occurrence");
           OpImportance = GetValue(Operation, "importance");
           If(OpOccurance == "Canceled" OR OpOccurance =="Not Done" ) AND
                OpImportance = "Obligatory") then
                    MessageBox("You should do this operation sins it is obligatory");
                    ExitLoop();
        }
        CallInference(Suggest);
        CallInference(Assign);
        SaveDoneOperationInPlantCareDB();
        LastSuggestedOp = GetLastSuggestedOperation();
        ClearListBox(SuggestedOperationListBox)
        FillListBox(SuggestedOperationListBox, LastSuggestedOp)
        DOTransfereTask(ProposedOperationDialogue);
End
```

On Double Click of Plant Status List Box.

```
Begin
        PlantStatus = GetListBoxText(PlantStatusListBox);
        SetToWM("plant-status", PlantStatus);
End
```

On Change Selection of Proposed Operation List Box

```
Begin
        SuggestedOp = GetListBoxText(SuggestedOperationListBox);
        OpStatus = GetValue(SuggestedOp, "occurrence");
        Case OpStatus{
                "NotDone": SetOpNoDoneRadioButtonStatus(TRUE);
                "Done" :  SetOpDoneRadioButtonStatus(TRUE);
                "Canceled": SetOpCanceldRadioButtonStatus(TRUE);
End
```

On Change Selection of Event List Box

```
Begin
        Event = GetListBoxText(EventListBox);
        EventStatus = GetValue(Event, "occurrence");
        Case EventStatus{
                "NotDone": SetEventNoDoneRadioButtonStatus(TRUE);
                "Done" :  SetEventDoneRadioButtonStatus(TRUE);
End
```

On Select Operation Done Radio Button

```
Begin
        SuggestedOp = GetListBoxText(SuggestedOperationListBox);
        SetToWM(SuggestedOp," occurrence", "Done");
End
```

On Select Operation Not Done Radio Button

```
Begin
        SuggestedOp = GetListBoxText(SuggestedOperationListBox);
        SetToWM(SuggestedOp," occurrence", "Not Done");
End
```

On Select Operation Cancel Radio Button

```
Begin
        SuggestedOp = GetListBoxText(SuggestedOperationListBox);
        SetToWM(SuggestedOp," occurrence", "Canceled");
End
```

On Select Event Done Radio Button

```
Begin
        Event = GetListBoxText(EventListBox);
        SetToWM(Event ," occurrence", "Done");
End
```

On Select Event Not Done Radio Button

```
Begin
        Event = GetListBoxText(EventListBox);
        SetToWM(Event ," occurrence", "Not Done");
End
```

On Click Previous Done Operation Button

```
Begin
        DOTransfereTask(DoneOperationDialogue);

End
```

On Click Exit Button

```
Begin
        CloseDialoge(PlantCareDialoge);
End
```

4 Transfer task "*Proposed Operation Dialogue*":

This section describe the design of the proposed operation dialogue and the event that are handled in this dialogue and the algorithms associated with each event

4-1 Dialogue Design

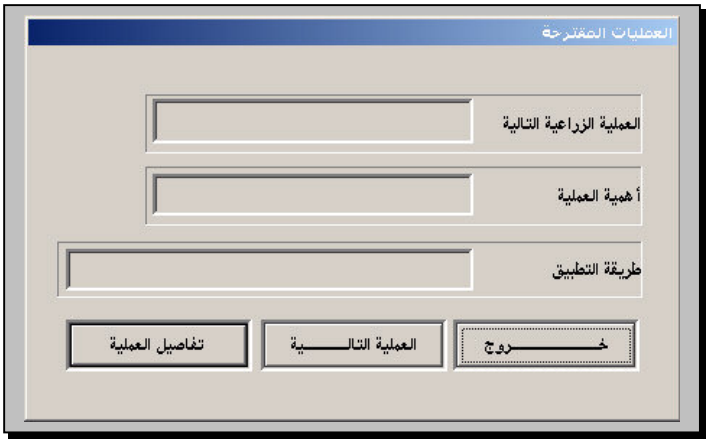

Figure 3

Figure 3 shows the design of the proposed dialogue, which consists of three text box, and three buttons.

4-2 Event Used in Proposed Operation dialogue:

The following are the events which is used in the proposed operation dialogue:

- On Initiate Dialogue.
- On Click Next Operation Button.
- On Click Operation Detail Button.
- On Click Exit Button.

4-3 Algorithms Associated With the Event:

The following algorithms are the event handler associated with the above events.

On Initiate Dialoge

```
Begin
        SuggestedOpList = GetSuggestedOperationFromWM();
        Define Global Pos = 0;
        Operation = GetFromList(SuggestedOpList, pos);
        If(Not Empty(Operation)){
                SetTextBoxData(MethodTextBox, Operation->method);
                SetTextBoxData(ImportanceTextBox, Operation->importance);
                SetTextBoxData(OpeartionNameTextBox, Operation->name);
        }
End
```

On Click Next Operation Button.

```
Begin
        Pos++;
        Operation = GetFromList(SuggestedOpList, pos);
        If(Not Empty(Operation)){
                SetTextBoxData(MethodTextBox, Operation->method);
                SetTextBoxData(ImportanceTextBox, Operation->importance);
                SetTextBoxData(OperationNameTextBox, Operation->name);
        }
        Else
                MessageBox("There is no other operation")
End
```

On Click Operation Detail Button.

```
Begin
        OperationName = GetTextBoxData(OperationNameTextBox);
        Operation = LookupInSuggestedOperation(OperationName)
        SndToDialog (Operation Detail Dialogue, Operation)
        If(Not Empty(Operation))
                DOTransfereTask(Operation Detail Dialogue);
End
```

On Click Exit Button.

```
Begin
        CloseDialogue(proposed operation dialogue)
End
```

5 Transfer task "*Operation Detail Dialogue*":

This Section describe the design of the operation detail dialogue and the event that are handled in this dialogue and the algorithms associated with each event

5-1 Dialogue Design

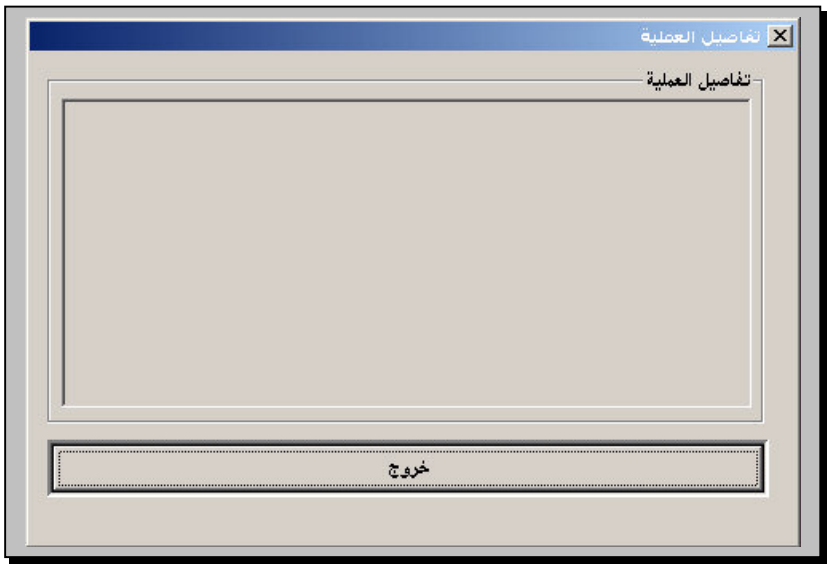

Figure 4

Figure 4 shows the design of the operation detail dialogue, which consists of one text box, and one button.

5-2 Event Used in Operation Detail dialogue:

The following are the events which is used in the operation detail dialogue:

- On Initiate Dialogue.
- On Click Exit Button.

5-3 Algorithms Associated With the Event:

The following algorithms are the event handler associated with the above events.

On Initiate Dialoge

```
Begin
        SetTextBoxData(DescriptionTextBox, Operation->Description);
End
```

On Click Exit Button

```
Begin
        CloseDialogue(Operation Detail Dialogue)
End
```

6 Transfer task " *Previous Operation Dialogue*":

This Section describe the design of the previous operation dialogue and the event that are handled in this dialogue and the algorithms associated with each event

6-1 Dialogue Design

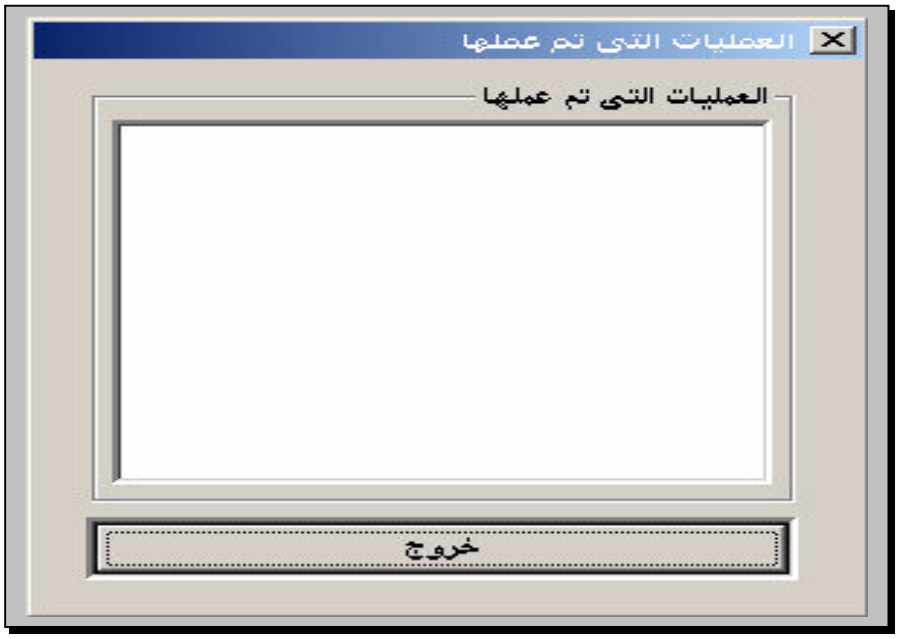

Figure 5

Figure 5 shows the design of the previous operation dialogue, which consists of one list box, and one button.

6-2 Event Used in Previous Operation dialogue:

The following are the events which is used in the previous operation dialogue:

- On Initiate Dialogue.
- On Click Exit Button.

6-3 Algorithms Associated With the Event:

The following algorithms are the event handler associated with the above events.

On Initiate Dialoge

```
Begin

        Operationslist = GetFromWM("Operations");
        For(int I = 0; I< GetCount(Operationslist); I++) Do
        {
                OpStatus = GetValue(Operationslist(I), "occurrence")
                If (OpStatus == "Done")
                        AddListBoxItem(OpListBox, Operationslist(I)->Name)
        }
End
```

On Click Exit Button

```
Begin
        CloseDialogue(previous operation dialogue)
End
```

7-Conclusion

The above template of the plant care subsystem which includes interface design and tasks, is an event driven Concept and the designer may follow this steps to accomplish its interface and task design.

- Draw Dialog.
- Add different control specification.
- Determine event to be handled on each control.
- Write association method