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

Detail Design for Treatment task Template

TR/CLAES/223/2001.7

By

Eng.Mohammed El Helly
Eng. Mohammed Yehia

July , 2001

1- Introduction

The objective of this report is to describe the template design of the interface, which is used in the Treatment subsystem. Also the event handlers associated with each interface component are described in more details. This report consists of two Sections the Interface Design and Events in Section 2 and transfer task "*LastMaterialUsageDialoge*" in Section 3, transfer task "*OptionalMaterialDialoge* in Section 4, Algorithms for order operation in Section 5, transfer task "treatment output" in section 6, and Conclusion in Section 7

2- Interface Design and Events

The Treatment subsystem is invoked from diagnosis subsystem by pressing the treatment push button in the following Figure

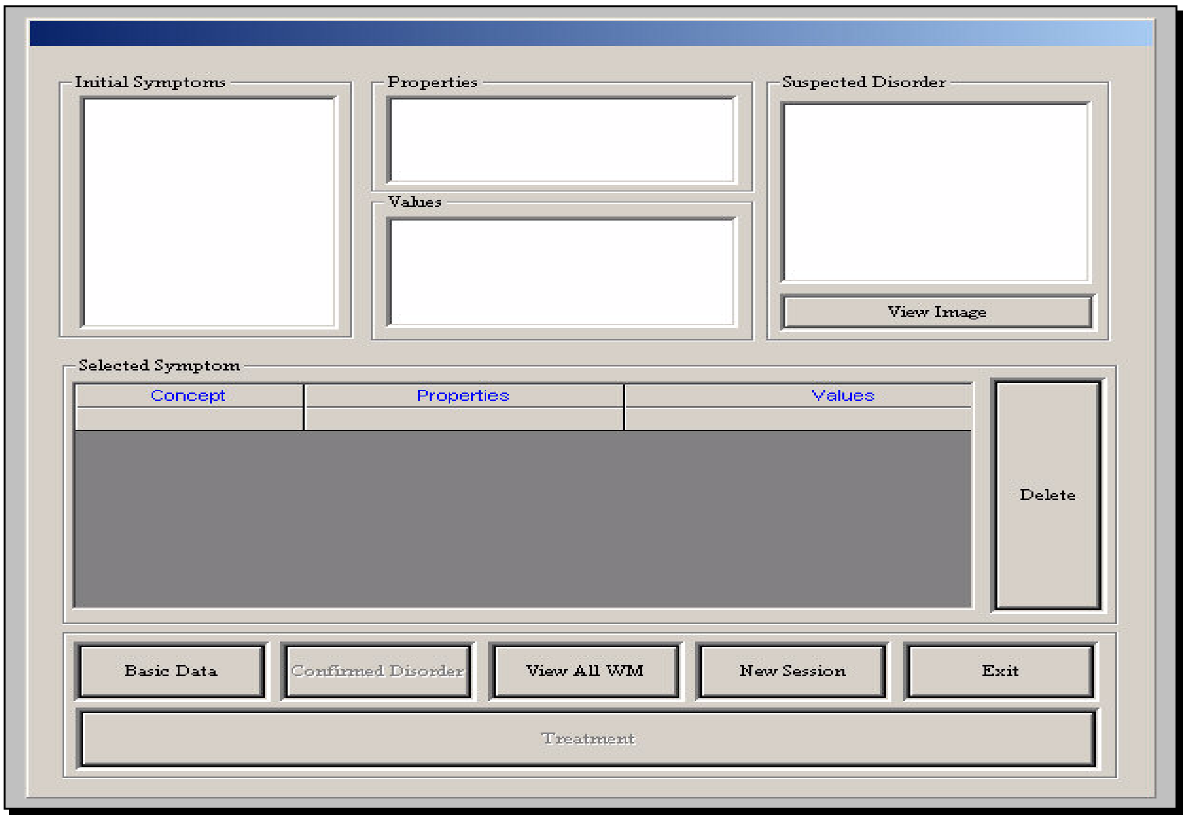


Figure 1

2-1 Main Task of the Treatment

It should be remarked that, this treatment button is enabled only if there exist at least one confirmed disorder in the disorder list box. The main task associated the treatment button when click event occur:

On Click Treatment Button.

```
Begin
        CallInference(SetDisorderPriority)
        CallInference(SetOpligatoryOptionalMaterial);
        DOTransfereTask(LastMaterialUsageDialoge)
        OptionalMaterialExist = GetValueOptionalMaterial();
        If(OptionalMaterialExist){
                DOTransfereTask(OptionalMaterialDialog);
        }
        CallInference(SetMaterialQTY);
        CallExpandInference(SetAppTime,Cpt-Prop, GenericCpt)
        CallExpandInference(SetAdvise,Cpt-Prop, GenericCpt)
        CallExpandInference(SetModeEntry,Cpt-Prop, GenericCpt)
        CallExpandInference(SetMaterialTool,Cpt-Prop, GenericCpt)
        CallSubTask(OrderOperation)
        DOTransfereTask(TreatmentOutputDialoge);
End
```

The above algorithm describe three transfer tasks that display the interaction dialogue with the user and one subtask which order the treatment operation according to disorder priority and difference between operations. We shall describe each of them individually in more details in the following sections.

3 Transfer task "*LastMaterialUsageDialoge*":

This Section describe the design of the last material used dialogue and the event that are handled in this dialogue and the algorithms associated with each event

3-1 Dialogue Design

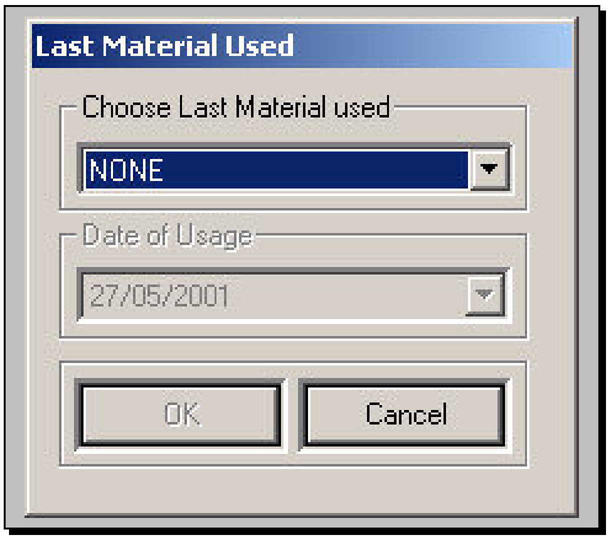


Figure 2

Figure 2 shows the design of the last material dialogue, which consists of material combo box, date of usage combo box, ok button, and cancel button.

3-2 Event Used in Last Material dialogue:

The following are the events which is used in the last material dialogue:

- On Initiate Dialogue.
- On Change Selection of Material Combo Box.
- On Click OK Button.
- On Click Cancel Button

3-3 Algorithms Associated With the Event:

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

On Initiate Dialoge

```
Begin
        SetDateComboBoxToNow ()
        MaterialList = GetMaterialsNameFromKB();
        FillMaterialsComboBox(MaterialList)
        AddComboItem(MaterialCombo, "NONE");
        SelectComboItem(MaterialCombo,"NONE");
        DisableOKButton();
        DisableDateComboBox();
End
```

On Change Selection of Material Combo Box

```
Begin
        MaterialName = GetCurrentSelectionText(MaterialComboBox);
        if(MaterialName == "NONE"){
                DisableOKButton();
                DisableDateComboBox();
        }
        else{
                EnableOKButton();
                EnableDateComboBox();
        }
End
```

On Click OK Button

```
Begin
        LastMaterialName = GetCurrentSelectionText(MaterialComboBox);
        DateOfUsage = GetDateFromDateCombo(DateOfUsageComboBox)
        if ((Day(DateOfUsage) – Day(Now) <= 15) OR (Month(DateOfUsage) – Month(Now) == 1)){
                RemoveMaterialFromObligatorySet(LastMaterialName)
        }
        CloseDialoge(LastMaterialUsageDialoge);
End
```

On Click Cancel Button

```
Begin
        CloseDialoge(LastMaterialUsageDialoge);
End
```

4 Transfer task “*OptionalMaterialDialoge*

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

4-1 Dialogue Design

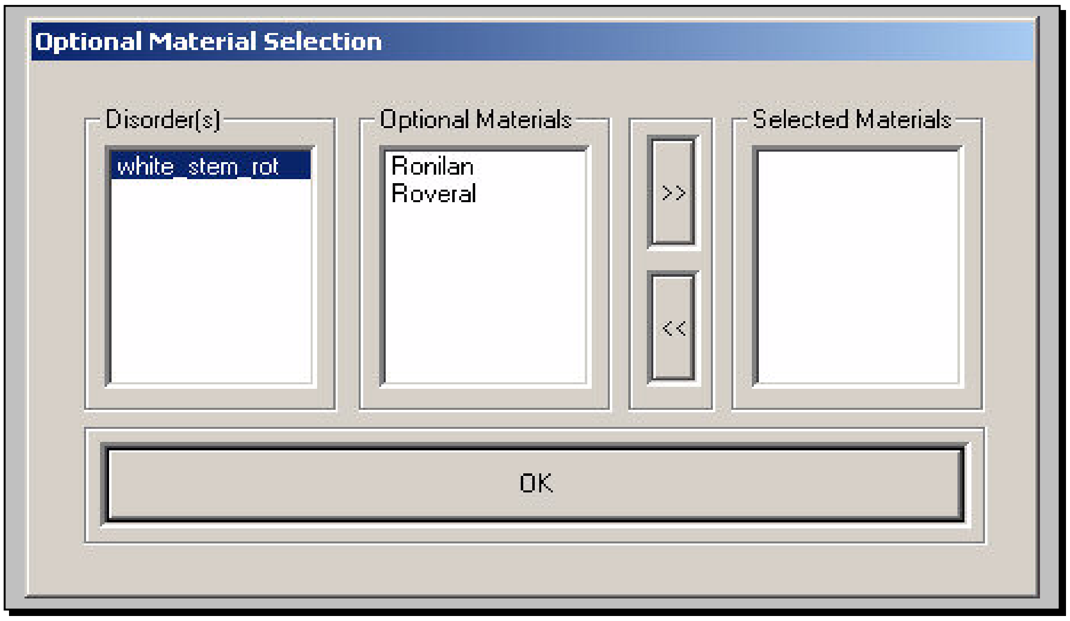


Figure 3

Figure 3 shows the design of the optional material selection dialogue, which consists of disorder list box, optional material list box, selected material list box, in button (>>), out button (<<), and ok button.

4-2 Event Used in Optional Material dialogue:

The following are the events which is used in the optional material dialogue:

- On Initiate Dialogue.
- On Change Selection of Disorder List Box.
- On Click In (>>) Button
- On Click Out (<<) Button
- On Click OK Button.

4-3 Algorithms Associated With the Event:

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

On Initiate Dialogue

```
Begin
        DisordersList = GetValue("OptionalMaterial","DisorderName");
        FillDisorderListBox(DisordersList);
End
```

On Change Selection of Disorder List Box.

```
Begin
        DisorderName = GetCurrentSelectionText(DisorderListBox);
        ClearListBox(OptionalMaterialListBox)
        OptionalMaterial = GetValue(DisorderName ,"OptionalMaterial");
        FillDisorderListBox(OptionalMaterialListBox);
End
```

On Click In (>>) Button

```
Begin
        DisorderName = GetCurrentSelectionText(DisorderListBox);
        OptionalMaterialName = GetCurrentSelectionText(OptionalMaterialListBox);
        ClearListBox(OptionalMaterialListBox);
        If(NOT(Find(OptionalMaterialName, SelectedMaterialListBox) then
                AddListBoxItem(SelectedMaterialListBox, OptionalMaterialName)
        End If
        Define Global StringList  DeletedDisorderList;
        DeletedDisorderList.SetAt(OptionalMaterialName, DisorderName);
        RemoveListBoxItem(DisorderListBox ,DisorderName)
        Integer index = 0;
        String fmdis;
        while((GetCount(DisorderListBox.)>0) && index < GetCount(DisorderListBox.)){
                fmdis  = GetIndexText(DisorderListBox, index);
                StringList OpList;
                OpList  = GetValue(fmdis,"OptionalMaterial");
                String strDis;
                If(GetCount(OpList >0){
                        If(Find (OpList, OptionalMaterialName)){
                                RemoveListBoxItem(DisorderListBox, index)
                                strDis = strDis +"@" +fmdis;
                                DeletedDisorderList.SetAt(OptionalMaterialName, strDis);
                        }
                        else index++;
                }
        }
End
```

On Click Out (<<) Button

```
Begin
        SelectedMaterialName = GetCurrentSelectionText(SelectedMaterialListBox);
        RemoveListBoxItem(SelectedMaterialListBox, SelectedMaterialName)
        DisordersList  = DeletedDisorderList. GetAt(SelectedMaterialName)
        If(DisordersList.GetLength() > 1 ){
                DisordersArray = Split(DisordersList,'@')
                For(i=0; i<DisorderArray.GetSize();i++)
                        If(DisorderListBox.Find( DisorderArray[i])){
                                AddListBoxItem(DisorderListBox, DisorderArray[i])
                                SetCurrentSelection(DisorderListBox, DisorderArray[i])
                        }
        }
        Else {
                AddListBoxItem(DisorderListBox, DisordersList)
                SetCurrentSelection(DisorderListBox, DisordersList)
        }
        SendMessage  (OnChangeSelectionOFDisorderListBox.)
End
```

On Click OK Button

```
Begin
        If(DisorderListBox.GetCount() != 0){
                MessageBox("You should select material for all disorders");Exit;
        }
        For(i = 0;i< SelectedMaterialListBox.GetCount();i++){
                SelectedMaterialName = GetListBoxText(SelectedMaterialListBox,i);
                If(! IsInWM("obligatory_material-name", SelectedMaterialName)
                        SetToWM("obligatory_material-name", SelectedMaterialName)
        }
        CloseDialoge(SelectionMaterialDialoge);
End
```

5-Order Operation

The order operation is responsible for rearrangement the produced treatment operations according two rules

1- The difference between two foliar or foliage operations at least 3 days.
2- The treatment operations should be sorted according t the priority of disorders.

```
Begin
        OpearationList = GetFromWM("obligatory_material" , "name");
        List = SortAccordingToDate(OpearationList);
        OrderdList = Order(List);
End
```

```
Function Order (List)
Begin
        BOOL satisfied = FALSE;
        while(!satisfied){
                for(int i =0;i<List.GetSize()-1;i++){
                        String M1 = GetValue(List[i],"method");
                        if(! (M1 == "foliar application" || M1 == "foliage nutrition")) continue;
                        for(int j=i+1;j<List.GetSize();j++){
                                String M2 =GetValue(List[j],"method");
                                if(! (M2 == "foliar application" || M1 == "foliage nutrition")) continue;
                                if(Satisfy3days(List[i],List[j]))break;
                                adjust(List[i],List[j],i,j,List);
                                List  = SortAccordingToDate(List);
                        }
                }
                satisfied = CheckSatisfy(List);
        }
End
```

```
Function CheckSatisfy(List)
Begin
        for(i =0;i<List.GetSize()-1;i++){
                        String M1 = GetValue(List[i],"method");
                        if(! (M1 == "foliar application")) continue;
                        String M2 = GetValue(List[i+1],"method");
                        if(! (M2 == "foliar application")) continue;
                        if(!Satisfy3days(List[i],List[i+1])){
                                return FALSE;
                        }
        }
        return TRUE;
End
```

```
Function adjust(Obj1, Obj2, i, j, List)
Begin
        TimeSpan  tsFinalDif, tsDif,ts(3,0,0,0) ;
        int ret;
        String D1 = r1->GetValue(Obj1,"date");
        String D2 = r1->GetValue(Obj2,"date");
        Time tD1 = r1->StrToDate(D1,ret);
        Time tD2 = r1->StrToDate(D2,ret);
        tsDif = tD1-tD2;
        tsFinalDif = ts- tsDif;
        int dif = tsFinalDif.GetDays();
        dif = abs(dif);
        String d1,d2;
        d1 =  GetValue(List[i],"disorder_name");
        d2 =  GetValue(List[j],"disorder_name");
        if(d1==d2) return;
        int p1 =GetValue(d1,"priority");
        int p2 =GetValue(d2,"priority");
        if(p1 < p2)
                Propagate(List[j],dif,j,List);
        else
                Propagate(List[i],dif,i, List);
Begin
```

```
Function SortAccordingToDate (List)
Begin
        String D1,Min,Temp;
        for(i = 0;i<List.GetSize()-1;i++){
                Min = GetValue(List[i],"date");
                for(int j = i+1;j<List.GetSize();j++){
                        D1 =GetValue(List[j],"date");
                        if(LessThanDate(D1,Min)){
                                Min = D1;
                                Temp = List[j];
                                List[j] = List[i];
                                List[i] = Temp;
                        }
                }
        }
End
```

```
Function LessThanDate(D1, Min)
Begin
    Time tD1,tMin;
    int ret;
    tD1 =StrToDate(D1,ret);
    tMin = StrToDate(Min,ret);
    if(tD1 < tMin)
        return TRUE;
    else
        return FALSE;
End
```

```
Function Satisfy3days(Obj1, Obj2)
Begin
    String D1 = GetValue(Obj1,"date");
    String D2 = GetValue(Obj2,"date");

    Time tD1,tD2;
    int ret;
    tD1 = StrToDate(D1,ret);
    tD2 = StrToDate(D2,ret);
    TimeSpan ts = tD1 - tD2;
    if (abs(tsGetDays ()) >=3)
        return TRUE;
    return FALSE;
}
```

```
Function Propagate(Obj, dif, j, List)
Begin
    AddTime(Obj,dif);
    String d1,d2;
    d1 =  GetValue(List[j],"disorder_name");
    for(i = j+1;i<List.GetSize();i++){
        d2 =  GetValue(List[i],"disorder_name");
        if(d1 == d2)
            AddTime(List[i],dif);
    }
End
```

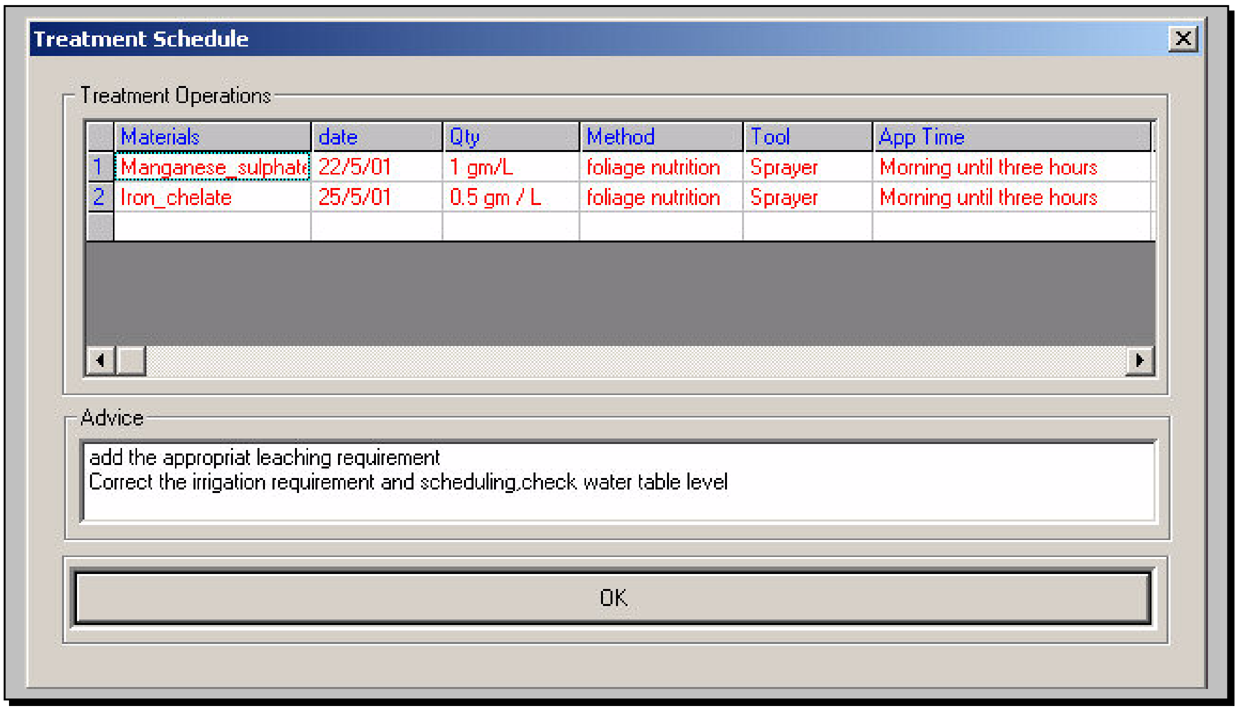
```
Function AddTime(Obj, dif)
Begin
        Time tD1;
        int ret;
        String D1 = GetValue(Obj,"date");
        tD1 = StrToDate(D1,ret);
        TimeSpan ts(dif+1,0,0,0) ;
        tD1 = tD1 + ts;
        SetToWM(Obj,"date", tD1);
End
```

6-Transfere Task "Treatment output

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

6-1 Dialogue Design

Figure 4 shows the design of the final dialogue, which consists of operations Flex Grid, Advice list box, , and ok button.



	Materials	date	Qty	Method	Tool	App Time
1	Manganese_sulphate	22/5/01	1 gm/L	foliage nutrition	Sprayer	Morning until three hours
2	Iron_chelate	25/5/01	0.5 gm / L	foliage nutrition	Sprayer	Morning until three hours

add the appropriat leaching requirement
Correct the irrigation requirement and scheduling,check water table level

Figure 4

6-2 Event Used in Final dialogue:

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

- On Initiate Dialogue.
- On Change Selection of Flex Grid.
- On Click OK Button.

6-3 Algorithms Associated With the Event:

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

On Initiate Dialogue

```
Begin
        OpearationsList = GetValue("Orderd Operations");
        FillFlexGrid(OpearationsList );
End
```

-On Change Selection of Flex Grid

```
Begin
        MaterialName = FlexGrid.GetRowText();
        ClearListBox(AdviceListBox);
        AdvicesList = GetValue(MaterialName,"Advice");
        FillAdviceListBox(AdvicesList);
End
```

On Click OK Button

```
Begin
        CloseDialoge(FinalDialoge);
End
```

7-Conclusion

The above template of the treatment subsystem 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