Last changed by:

2010-11-01

**Documentation of HotLine Support System**

**© Soren Lauesen**

**Contents**

[1. Background 1](#_Toc231382996)

[2. Time wasters and lessons learned 1](#_Toc231382997)

[3. Introduction to the system 6](#_Toc231382998)

[4. Screens 7](#_Toc231382999)

[5. basCommon 8](#_Toc231383000)

[6. frmRequest 9](#_Toc231383001)

[7. frmRequestList 13](#_Toc231383002)

[8. fsubRequestList 15](#_Toc231383003)

[9. frmSupporters 15](#_Toc231383004)

[10. frmEmail 15](#_Toc231383005)

[11. frmNow 16](#_Toc231383006)

[12. Test data and test environment 17](#_Toc231383007)

[13. Defects 17](#_Toc231383008)

[14. Test cases 18](#_Toc231383009)

# Background

The Hotline Case started out as a written exam in User Interface Design, January 2006. I knew several small hot-line organizations, and I studied the one at ITU in detail to learn about their work and problems. This was the basis for the exam text. The students were asked to define the user tasks to support, the data model, the virtual windows, and the functions on each window. As usual, I had made a reply myself to calibrate the difficulty of the assignment (version 1).

The next semester, the exam text was used for training. After handing in their reply, the students got my reply. The students should now construct and program the system in MS Access. Most students managed to construct the user interface, the database, and connect them. However, they didn't get far with the programming for several reasons, one of them being that my solution looked easy from an MS Access point of view, but wasn't.This is explained further in section 2.

One semester later, we talked about using the exercise again and maybe do the programming in Java. I revised my reply (version 2) to make it easier to implement in MS Access as well as Java. Then I developed the MS Access version myself to calibrate the difficulty. Although much easier this time, there were still many challenges. This report describes the result and the lessons learned. The Java version was never attempted.

The original exam text and reply version 2 are included in the zip-file as file DBDexam05A.doc and DBDreply05A2.doc.

# Time wasters and lessons learned

The virtual windows in version 1 and 2 are shown below. Both are very close to the final MS Access user interface shown in section 4. The main difference is that version 1 allowed creation and some editing of requests directly in the overview, the **request list**. It was designed this way because hot-line staff should always record hot-line requests, even the simplest ones such as giving the user a new password, but they didn't because it was too cumbersome. The idea was that all the frequent small requests could be handled in the list with a few keystrokes.

Constructing Access windows that looked like version 1 was simple. However, making them work was a nightmare. The basic problem was that while editing a new request in the list, the supporter could easily move the cursor up or down, or click somewhere else in the window. This caused the partially edited request to be saved to the database and logged in the history record, probably in an inconsistent state.

Reply version 2 didn't allow editing in the list. The supporter had to create or open a full request window. However, the layout and functions in this window made recording fast. For instance it is possible to record and close simple requests such as "forgotten password for user ann3" with 10 keystrokes. The original solution would need 8 keystrokes. I believe that the revised solution is not only vastly easier to implement, it is also much easier to understand for the user (the supporter). However, I haven't tested the understandability.





Lesson 1: A tool-based mockup doesn't ensure programmability

Reply version 1 seemed easy to make in Access. It looked right, but programming the event handlers was a night-mare. The controls wouldn't behave as I had specified in the function design.

Lesson 2: Final solution very close to design

Reply version 2 deviates very little from the actual solution in section 4. The main differences are addition of one more search criterion to the request list, rearranging the fields in the full request window a bit, and correcting minor errors in the detailed functional specification. Thus the virtual window approach proved useful in this case.

Lesson 3: Spend time early making a good test environment

I spent too much time fiddling around with ad-hoc data base contents that were changed during the test and hard to repeat. The reason I didn't define the test data early on was probably that I mentally grabbled with how the system would look in real-life situations with lots of data, and whether users could use it intuitively. The lesson was that this had to be two different sets of test data. The long one would be suited for usability testing, but not for program testing.

I had planned a simulated environment where the clock was set manually, the current supporter was specified, and test mode was set or cleared. The Now window handled this in a primitive way. However, development speeded up when I introduced a Reset Data button. It deleted records in the operational database (tblRequest, tblHistory and tblSupporter) and inserted fresh values from tblRequestInit, tblHistoryInit and tblSupporterInit. I also removed the Update button from the Now window, because I often forgot to use it. Instead I auto-updated whenever some field in Now was changed.

Lesson 4: Beware of recursive events

The design suggested that the supporter didn't have to click an Edit button when adding information to a request. He should just start changing what he wanted to change. The Form's Dirty event would be triggered when he did so. The latest note about the request was shown in the request form, and when edit started, the note space had to be cleared to make space for a new note. (The entire trail of notes was visible in the history list at the bottom of the Form.)

It sounded simple: At the Dirty event, simply clear the note field (and make other small things such as showing the current supporter as source of the changes). This seemed to work sometimes, but failed now and then in a random fashion. In a few days, I found out that Dirty was a recursive event. When another field was changed in the event procedure, Dirty was called again. Access didn't report any stack overflow - it just acted randomly.

The cure was easy: Test at entry whether Dirty is active already. If so, return. If not, record that Dirty is active - and clear the record when Dirty returns in the normal way.

Lesson 5: Hard to let user and program write in the same field

I had expected that the use of the Note field to show either the latest note or the new note, might be confusing to the user. The design in the reply suggested a variable label to the note field, saying either *Latest Note* or *New Note*, but I knew that users might not notice it. So I tried to write *(New Note)* in the Note field itself, in the same fashion as you often see an explanatory text in data fields on the web. However, I couldn't find a way to make the program remove the explanatory text when the user started typing in the note field. Events interacted in obscure ways.

I ended up using the suggested solution in the design, but changed the background color of the label when it showed *New Note* in order to make the change more visible.

Lesson 6: Display-update is obscure

I have struggled several times to make Access update the display when the program changes something. Several techniques are available: Repaint, Requery, Refresh, setting the control rather than the bound database field - or vice versa. What works seems to depend on whether the control is a simple textbox, a bound one, a computed one, a combobox, etc. A consistent description of the rules would have saved days. And as usual, you couldn't trust the on-line documentation.

Lesson 7: When is a blank text Null?

The basic Null rules in Visual Basic are very logical. Null means "any value". As a result 3+Null is Null, Null And True is Null, etc. Cleverly, Null And False is **False**, however.

The problem is blank texts. In the database, a blank text field is Null, but in a VBA string variable it is "". Controls seem to use "" for their own value, even though the database field they are bound to is Null. Combo boxes with the property *Limit to List* often allow Null - even though Null isn't in their list of values. To test for a blank text, the program has to ask for t = "" or IsNull(t) depending on what kind of t we deal with. A mistake is not detected until the situation occurs at run time. A consistent description of the rules would have saved a day of work.

Lesson 8: Access often forces a database update on its own initiative

The main problem with my first design was that data would be committed before the user wanted it. If he edited a row in a table and moved the cursor up or down, data would be committed. Surprisingly, the new design had similar problems, but smaller. When a main form is bound to a database record, and the user clicks a subform or a tabsheet on the same form, the main-form data will be committed. There seems to be no way around it. I believe it will be necessary to program the true semantic transaction by means of auxiliary tables that are transferred to the database at carefully programmed points. Or maybe it is possible to use the database commit and rollback functions, which normally aren't necessary.

In the hot-line system, the problem only occurred in the Request window when the user moved to the history list. If this list was a subform, the hard solution would be necessary. I avoided the problem by making one of the history views a List control. It didn't force database updates. The other view was one big text control, computed by the program. (Changing between the two views by hiding one of the controls was easy and it became a nice piece of program. Here Access worked as expected.)

Lesson 9: Tricky to stop closing a create-window

The BeforeUpdate event is intended to check data before it is committed. The event procedure can stop the update by setting Cancel to True at return. The user will then have to adjust the data - or use Esc to reset data to the initial values - before he can close the window.

Unfortunately this doesn't work if the Form handles creation of a new record, and the user closes the window. BeforeUpdate may find an error, show the user a message, and cancel the update in order that the user can correct the data. However, Access will interfere and ask the user in strange terms whether he wants to store the record - Yes or No. The program may catch such errors in the OnError event before the question, but it cannot stop Access's strange question. It took me two days to figure out how to deal with it: In the OnError procedure, use SendKeys("n") to send Access the letter N as if the user had typed it for *No*. It works so fast that the user cannot see the question at all.

Lesson 10: Setting one of the Form's CRUD attributes may change other settings as well

When the program opens a form, it specifies whether it is for editing, creation, etc. However, the user may still tab from one record to the next, go to the end and create a record, or delete a record. We don't want to allow this in the hot-line system, but the programmer may not notice the problem except by chance.

Some programming is needed to avoid it. All the CRUD attributes (AllowAdditions, AllowDeletions, DataEntry, etc.) must be set from the program. If a new request is created and saved, but the window remains open, the CRUD attributes must be set to reflect it. An apparently unrelated attribute, Cycle, must also be set to avoid that tabbing past the end of the record saves the record prematurely. A particularly annoying thing is that setting DataEntry to False has a strange side effect on the other CRUD attributes, which may cause Access to select a completely unintended record. It took another day to figure this out - only to realize that I had discovered it before and even explained it in my own Access book about two years earlier. In the book I also gave the solution: Set DataEntry first, then set the other attributes.

Lesson 11: Times must be exact fractions, and Input Masks don't work for date-time

Access and Visual Basic treat date-time as a double precision floating point number that specifies the number of days after 30th December 1899, 0:00. Time is represented as a fraction of a day. The hotline system shows the time as the number of hours and minutes, omitting the seconds. It often happened that when the user started editing the time, it suddenly showed the number of seconds too, apparently in a random fashion. Even more puzzling was that it sometimes showed a completely empty date and time when edit started. It seemed as if the system had completely erased the time.

Access offers a feature for setting up an input mask (a template) with a fixed number of input characters and separators. I tried to use it to lock the date-time format so that Access didn't display the seconds. It only made things worse. Seeking advice on the web only gave trivial information or misleading advice (as it often happens).

After three days of struggle, I found the answers: Input masks work okay for dates and ok for times, but not for a field that contains both, for instance 25-12-2007 11:35 with the format dd-mm-yyyy hh:nn.

Further, if a date-time isn't an exact number of seconds, Access assumes it is not a time but a number, and starts editing with a blank field. If it is an exact number of seconds but not an exact number of minutes, edit shows the seconds too. In my case, the program had computed the times based on the built-in clock and they were not an exact number of seconds.

Lesson 12: Hard to design a systematic test procedure

When testing a highly user-centered system, it is hard to define the test cases. In this case I had to test against the requirements in my own reply, the design specification in my reply, and the branches in the code. Although I knew this in theory, it took a day to develop the proper way to structure the test cases. In particular, although I had claimed that you can test whether a task is properly supported, I had never done it in practice.

The principle became this: Check that the system supports the data model. Check that it supports all tasks including variants. Test that it meets the design specification (okay to skip test cases that have been handled earlier). Test that all branches of the program have been taken (in our case the earlier test cases had covered most of the branches). Finally, we should usability test the system to see that it meets the usability requirements, but I have not done this yet.

Lesson 13: Many dialog states are invisible, but functionality may crash in some of these states

The user dialog may be in a "hidden" state. Testing the functionality in this state may thus be forgotten. In the hotline application several hidden states caused problems: When a new request is being created in the request form, the form doesn't behave as usual. This caused system crashes. When the request form is open, functions in the request list form don't behave as usual. This too caused crashes. The application includes certain libraries, but somehow this information is not always transferred when the application is moved to another computer.

Time spent

I have spent around 80 hours implementing and testing the system. Roughly half of the time was dealing with the oddities of Access (lessons 4-11). Roughly one quarter was developing the test, carrying it out and correcting the bugs. The errors found during the test were very easy to correct except where Access oddities were involved. The remaining quarter was construction and programming.

You may wonder why I several times above state that I struggled with a problem for two days. It would give a total of much more than 11 days. The answer is that I didn't work full time on the project, and usually I worked on several mysteries at the same time. Typically, one or two nights of sleep helped me solve a problem.

It should be noted that I am not an Access novice. I had written an Access tutorial that covered many things, including an in-depth treatment of event procedures and VBA. It had taken me around a year full time to uncover how Access actually works. Yet when developing something as simple as the hotline system, the Access oddities were still the largest time waster in the project.

# Introduction to the system

The hotline system consists of only three Forms (Access windows): Request list (frmRequestList), Request (frmRequest) and Supporters (frmSupporters). In addition two Forms serve as a simulated environment during testing: Now (frmNow) and Email simulator (frmEmail).

**Request list** is the supporter's main Form. It gives him an overview of what is to be done and allows him to search for requests, create new requests, open requests to see details, and open the list of supporters.

**Request** gives all the details of a single request, including the history of the request as it passes from one supporter to another.

**Supporters** show the state of all the supporters, whether they are on duty, and whether they want to receive email notification when another supporter assigns a request to them.

**Now** allows changing of the simulated time and the current supporter. In test mode, whenever the program calls getTime() to get the current time, the system increases the current time by one minute to simulate the passage of time. Now also allows changing from test mode to real mode. Finally, it has a button for resetting the tables to the contents used when a test case starts.

**Email simulator** shows an email being sent to another supporter or a user. In real mode the message would just be sent without showing it to the supporter. Before sending the mail, the system will usually ask the supporter whether he wants message mmm to be sent. The exception is when the message goes to another supporter who has asked for notification.

VBA code

In Access each Form is a class with local properties, e.g. data properties, local procedures (methods), and event handlers for all the controls on the Form. In addition the hot-line system has a baseCommon class with common service procedures and global variables. Sections 4 to 11 show the VBA program in each of the classes.

Access cannot print the VBA code with a nice layout, so I have copied the code manually, pasted it into the Word document with Paste Special, and added lines of underscores manually for visually separating the procedures.

Tables

According to reply version 2, the system uses these tables in real life mode:

tblRequest The requests in their latest state.

tblHistory The requests in their latest state and all previous states.

tblCause The possible causes that can be attached to a request. They form a hierarchy by means of a self-reference.

tblEmployee All employees who can get support. In the test database, employees who are supporters have an (S) indication at the end of their name to help the tester see what he is doing.

tblSupporter All supporters. In principle this is a subclass of the employees.

tblPriority A table of possible priorities. In principle an enumeration type would suffice, but in order to allow customization and language changes, a table is used.

tblStatus A table of possible states. For the same reason as tblPriority it is not an enumeration type.

To support testing, the system also has these tables:

tblRequestInit The seven requests used for testing.

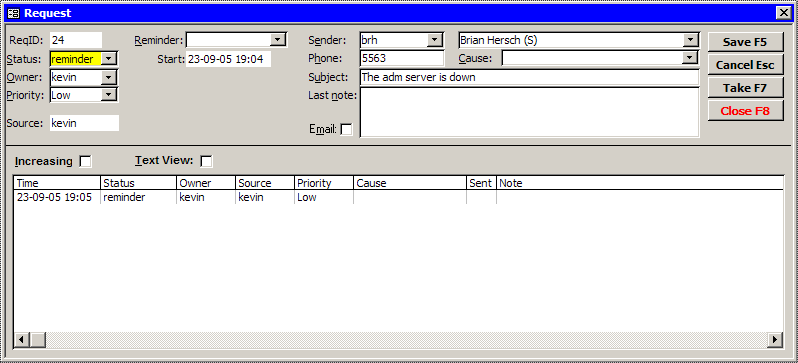
tblRequestInit2 A larger number of requests used for usability testing, etc. (Not completed.)

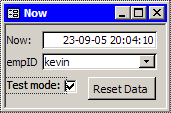
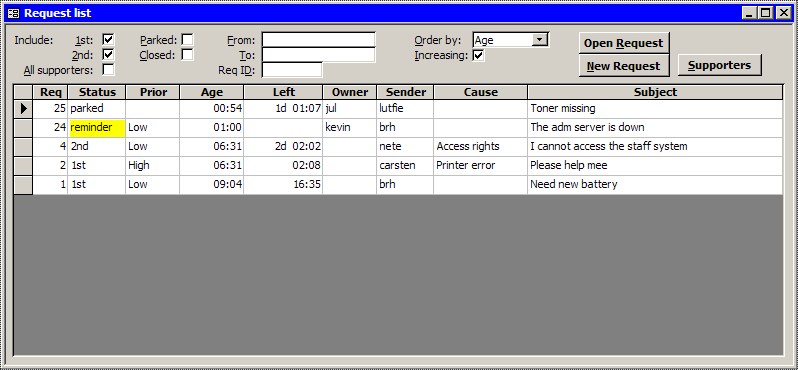
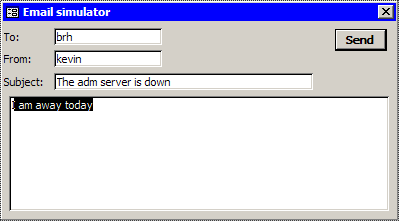
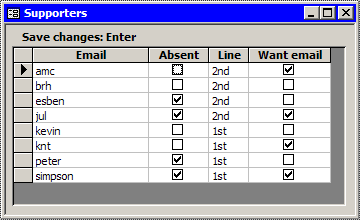
tblHistoryInit The seven requests used for testing in their initial state.

tblSupporterInit The supporters and their initial assignment to line 1 and 2, their absence, and their email preference.

tblNow One record holding current simulated time, current supporter and test mode.

# Forms





# basCommon

Option Compare Database

Option Explicit

Public Const c1st = 1, c2nd = 2, cTaken = 3, cParked = 4, cReminder = 5, cClosed = 6

Public testMode As Variant ' Initially Empty. At Reset during debugging, the current value is lost and testMode becomes Empty. GetTestMode will then retrieve it from tblNow.

' GetTime returns time as a Double. In testMode it uses Dlookup in tblNow rather than Now(). It also increases time by one minute to simulate passing af time.

' GetTimeSQL returns a string to be used in SQL. In testMode it returns a string with "Dlookup" rather than the string "Now()".

' This allows a new time to reflect in the result without having to recompute the SQL statement.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public parTo As Variant, parFrom As Variant, parSubject As Variant, parBody As Variant ' Call parameters for simulated email sending

Public mailRight As Integer, maildown As Integer ' For making the simulated email box move around a bit

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public Function Solid(v As Variant) As Variant ' Returns 0 or "" instead of Null

If IsNull(v) Then

Solid = 0

Else

Solid = v

End IfEnd Function

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public Sub fncSendMail(sender As String, subject As Variant, body As Variant)

parTo = sender

parFrom = DLookup("email", "tblEmployee", "empID = " & getSupporter())

parSubject = subject

parBody = body

DoCmd.OpenForm "frmEmail", , , , , acDialog

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public Function getTime() As Double

Dim s As String, t As Double

If getTestMode() Then

s = "UPDATE tblNow SET tblNow.dateTime = CDbl(dateTime) + 1 / 1440 WHERE id = 1;" ' Increase Now by 1 min for testing

CurrentDb.Execute s, dbFailOnError

t = DLookup("dateTime", "tblNow", "ID = 1")

s = "UPDATE tblRequest SET tblRequest.status = 5, tblRequest.reminderTime = Null WHERE tblRequest.reminderTime < " & t & ";" ' Set reminders

CurrentDb.Execute s, dbFailOnError

getTime = t

On Error Resume Next ' The forms may not be open

Forms!frmNow.Refresh

Forms!frmRequestList.Requery ' Requery should be enough since Current supporter hasn't changed.

Else

getTime = Now()

End If

End Function

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public Function getTimeSQL() As String

' To allow requeries, SQL will use Now() or Dlookup rather than a computed time, so that a new time will reflect in the result at requery

If getTestMode() Then

getTimeSQL = "DLookup('dateTime', 'tblNow', 'ID = 1')"

Else

getTimeSQL = "Now()"

End If

End Function

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public Function getSupporter() As Long

If getTestMode() Then

getSupporter = DLookup("empID", "tblNow", "ID = 1")

Else

' getSupporter = CurrentUser ' Returns "Admin". Should be the ID of the user who is logged in.

getSupporter = DMin("empID", "tblSupporter", "True") ' Replacement when testing testMode

End If

End Function

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public Function getTestMode() As Boolean

If IsEmpty(testMode) Then testMode = DLookup("testMode", "tblNow", "ID = 1")

getTestMode = testMode

End Function

# frmRequest

Option Compare Database

Option Explicit

Dim oldOwnerID As Variant ' Saved ownerID to be used in AfterUpdate to decide on message to send.

Dim activeDirty As Boolean ' Stops endless recursion in Dirty event

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub setHistoryList()

Dim s As String, h As String, rs As Recordset

If chkTextView Then

s = "SELECT qryHistory.Time, qryHistory.Status, qryHistory.Source, qryHistory.Sent, qryHistory.Note FROM qryHistory WHERE requestID = "

s = s & requestID & " ORDER BY qryHistory.Time "

If Not chkIncreasing Then s = s & "DESC "

Set rs = CurrentDb.OpenRecordset(s & ";")

While Not rs.EOF

h = h & Format(rs!Time, "dd-mm-yy hh:nn") & " " & rs!status & ". " & rs!source & IIf(IsNull(rs!sent), " noted: ", " sent: ") & rs!note & vbCrLf & vbCrLf

rs.MoveNext

Wend

Me.txtHistory = h

txtHistory.Visible = True

lstHistory.Visible = False

Else

s = "SELECT qryHistory.Time, qryHistory.Status, qryHistory.Owner, qryHistory.Source, qryHistory.Priority, qryHistory.Cause, qryHistory.Sent, qryHistory.Note FROM qryHistory WHERE requestID = "

s = s & requestID & " ORDER BY qryHistory.Time "

If Not chkIncreasing Then s = s & "DESC "

lstHistory.RowSource = s & ";"

txtHistory.Visible = False

lstHistory.Visible = True

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboCauseID\_AfterUpdate()

If IsNull(txtSubject) And cboCauseID Then txtSubject = DLookup("name", "tblCause", "causeID = " & cboCauseID)

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboOwnerID\_AfterUpdate()

If cboOwnerID > 0 Then cboStatus = cTaken

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboReminder\_AfterUpdate()

Dim t As Date

If cboReminder = 0 Then

cboReminder = Null

ElseIf cboReminder < 500 Then ' < 500 days: Selected from list

t = getTime() + cboReminder

cboReminder = CDate(Format(t, "mm-dd-yy hh:nn")) ' Must be an exact number of minutes to allow editing

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboSenderEmail\_AfterUpdate()

txtPhone = DLookup("phone", "tblEmployee", "empID = " & senderID)

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboSenderEmail\_BeforeUpdate(Cancel As Integer)

If IsNull(senderID) Then ' Access 2000 accepts blanks, but Access 2003 makes entire Access crash if you undo after setting a bound cbo to blank.

Call MsgBox("Sender cannot be blank", vbOKOnly)

Cancel = True

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboSenderName\_AfterUpdate()

txtPhone = DLookup("phone", "tblEmployee", "empID = " & senderID)

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboSenderName\_BeforeUpdate(Cancel As Integer)

If IsNull(senderID) Then

Call MsgBox("Sender cannot be blank", vbOKOnly)

Cancel = True

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboStatus\_AfterUpdate()

If IsNull(cboStatus) Then cboStatus = 1

If cboStatus = cClosed Then

cboReminder = Null

ElseIf cboStatus <= c2nd Then

cboOwnerID = Null

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkIncreasing\_AfterUpdate()

Call setHistoryList

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkTextView\_Click()

Call setHistoryList

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdCancel\_Click()

Me.Undo

If DataEntry Then

DoCmd.Close

Exit Sub

End If

lblNote.Caption = "Last &note:"

lblNote.BackColor = -2147483633

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdClose\_Click()

cboStatus = cClosed

cboReminder = Null

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdSave\_Click()

Me.Refresh

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdTake\_Click()

cboOwnerID = getSupporter()

cboStatus = cTaken

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_AfterUpdate()

Dim s As String, rs As Recordset

Dim emailSender As String, newSupporter As Long

Set rs = CurrentDb.OpenRecordset("SELECT \* FROM tblhistory;") ' Use record set for updating to avoid problems with Null, empty texts, etc.

rs.AddNew

rs!requestID = requestID

rs!saveTime = getTime()

rs!reminderTime = reminderTime

rs!subject = subject

rs!note = Me!note

rs!priority = priority

rs!currentPhone = currentPhone

rs!status = status

rs!causeID = causeID

rs!senderID = senderID

rs!sourceID = sourceID

rs!ownerID = ownerID

rs!sendMail = Me!sendMail

rs.Update

rs.Close

If Me.DataEntry Then ' Create request completed

Me.AllowAdditions = False

Me.Filter = "requestID = " & requestID

Me.DataEntry = False ' Sets FilterOn = False

Me.FilterOn = True ' Moves to first record if done before setting filter

End If

lblNote.Caption = "Last &note:"

lblNote.BackColor = -2147483633

If chkTextView Then setHistoryList

emailSender = DLookup("email", "tblEmployee", "empID = " & senderID)

If sendMail And txtNote <> "" Then Call fncSendMail(emailSender, subject, txtNote)

If ownerID <> 0 Then

If oldOwnerID = ownerID Or DLookup("empID", "tblEmployee", "empID = " & ownerID) = sourceID Or Not DLookup("notifyByEmail", "qrySupporter", "empID = " & ownerID) Then

' Same owner or myself or doesn't want notification

Else

Call fncSendMail(DLookup("email", "tblEmployee", "empID = " & ownerID), "Request " & requestID, "Look at request " & requestID & " from: " & emailSender & ". Subject: " & subject)

End If

End If

On Error Resume Next ' frmRequestlist may not be open

Forms!frmRequestList.lastRequest = requestID

Forms!frmRequestList.Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_BeforeUpdate(Cancel As Integer)

Dim result As Long, noteLineFeed As String

noteLineFeed = IIf(Me.txtNote = "", "", Me.txtNote & vbCrLf) ' Saves a line if the note is empty

oldOwnerID = cboOwnerID.OldValue ' Prepare sending mail to a new owner

If IsNull(senderID) Or IsNull(subject) Then ' Req 1: Sender and subject always required

Call MsgBox("Sender and subject must be defined." & IIf(IsNull(subject) And IsNull(cboCauseID), vbCrLf & "You may set cause instead of subject.", ""), vbOKOnly)

Cancel = True

If IsNull(senderID) Then

cboSenderEmail.SetFocus

ElseIf IsNull(cboCauseID) Then

cboCauseID.SetFocus

Else

txtSubject.SetFocus

End If

Exit Sub

End If

If Solid(cboStatus.OldValue) <> cClosed And cboStatus = cClosed And IsNull(cboCauseID) Then ' Req 2: Closed and no cause: Set one

If MsgBox("Set a cause, please", vbOKCancel) = vbOK Then

Cancel = True

cboCauseID.SetFocus

Exit Sub

End If

End If

If Solid(cboStatus.OldValue) <> cClosed And cboStatus = cClosed And (txtNote = "" Or Not sendMail) Then ' Req 3: Closed: Offer user message

result = MsgBox("Send this mail to user?" & vbCrLf & vbCrLf & noteLineFeed & "Your request number " & requestID & " has been closed. ", vbYesNoCancel)

If result = vbYes Then

Me.txtNote = noteLineFeed & "Your request number " & requestID & " has been closed. "

sendMail = True

ElseIf result = vbCancel Then

txtNote.SetFocus

Cancel = True

Exit Sub

End If

End If

If Solid(cboStatus) < cReminder And IsNull(cboReminder) Then ' Req 4: Not closed or reminder, and no reminderTime: Set one

If MsgBox("Set a reminder time?", vbOKCancel) = vbOK Then

cboReminder.SetFocus

Cancel = True

Exit Sub

End If

End If

If cboReminder <> 0 And Solid(cboReminder.OldValue) <> Solid(cboReminder) And (txtNote = "" Or Not sendMail) Then ' Req 5: New reminder: Offer user message

result = MsgBox("Send this mail to user?" & vbCrLf & vbCrLf & noteLineFeed & "Your request number " & requestID & " should be finished at " & Format(reminderTime, "dd-mm-yy hh:nn"), vbYesNoCancel)

If result = vbYes Then

Me.txtNote = noteLineFeed & "Your request number " & requestID & " should be finished at " & Format(reminderTime, "dd-mm-yy hh:nn")

sendMail = True

ElseIf result = vbCancel Then

txtNote.SetFocus

Cancel = True

Exit Sub

End If

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_Dirty(Cancel As Integer)

Dim s As Long

If activeDirty Then Exit Sub ' Endless calls of Dirty may take place when sourceID is set during Dirty.

activeDirty = True

lblNote.Caption = "New &note:"

lblNote.BackColor = 16777215 ' White

txtNote = Null ' Strangely, this becomes "" and IsNull gives False

sourceID = getSupporter() ' Updates txtSource automatically when it is a computed control.

activeDirty = False

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_Error(DataErr As Integer, Response As Integer)

If DataErr = 2169 Then SendKeys "n" ' Answer No to "Close this object?"

' When the user closes a new request window and BeforeUpdate finds an error, it is not suffient that BeforeUpdate cancels

' the update. Access will trigger DataErr 2169, and even if we return 0 as response, Access will ask in cryptic words

' whether the user wants to save the record - Yes or No. SendKeys answers No on behalf of the user.

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_KeyDown(KeyCode As Integer, Shift As Integer)

If KeyCode = vbKeyF5 And Shift = 0 Then ' Save

Call cmdSave\_Click

KeyCode = 0

ElseIf KeyCode = vbKeyF7 And Shift = 0 Then ' Take

Call cmdTake\_Click

KeyCode = 0

ElseIf KeyCode = vbKeyF8 And Shift = 0 Then ' Close

Call cmdClose\_Click

KeyCode = 0

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_KeyUp(KeyCode As Integer, Shift As Integer)

If KeyCode = vbKeyEscape And Not Dirty Then ' Necessary in Access 2000. In Access 2003, the Undo event can reset the caption.

If DataEntry Then ' New request

DoCmd.Close

Exit Sub

End If

lblNote.Caption = "Last &note:"

lblNote.BackColor = -2147483633

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_Load()

Dim s As String

If DataEntry Then ' Create new request

sourceID = getSupporter()

cboStatus = c1st

txtStartTime = getTime()

chkEmail = False

cboSenderEmail.SetFocus

lblNote.Caption = "New &note:"

lblNote.BackColor = 16777215 ' White

Else

AllowAdditions = False

lblNote.Caption = "Last &note:"

lblNote.BackColor = -2147483633

End If

AllowDeletions = False

chkIncreasing = False

chkTextView = False

txtHistory.Visible = False

Call setHistoryList

End Sub

# frmRequestList

Option Compare Database

Public oldFocus As String

Public lastRequest As Double ' This request will always be shown in the list to provide visual feedback about request changes

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Public Sub Search()

Dim s As String

s = "SELECT tblRequest.requestID, tblStatus.statusName, owner.email AS qOwner, tblPriority.priorName, "

s = s & getTimeSQL() & "-startTime AS ageD, remindertime-" & getTimeSQL() & " AS leftD, "

s = s & "Iif (ageD < 0, '-', ' ') & Iif(int(Abs(ageD))=0,' ', format(int(Abs(ageD)),'0\d ')) & format(Abs(ageD)-int(Abs(ageD)),'hh:nn') AS qAge, "

s = s & "Iif (leftD < 0, '-', ' ') & Iif(int(Abs(leftD))=0,' ', format(int(Abs(leftD)),'0\d ')) & format(Abs(leftD)-int(Abs(leftD)),'hh:nn') AS qLeft, "

s = s & "sender.email AS qSender, tblCause.name AS qCause, tblRequest.subject "

s = s & "FROM (((( tblRequest LEFT JOIN tblemployee AS sender ON sender.empID=tblRequest.senderID) "

s = s & "LEFT JOIN tblemployee AS owner ON owner.empID=tblRequest.ownerID) "

s = s & "LEFT JOIN tblStatus ON tblRequest.status=tblStatus.statusID) "

s = s & "LEFT JOIN tblPriority ON tblRequest.priority=tblPriority.priorID) "

s = s & "LEFT JOIN tblcause ON tblCause.causeID=tblRequest.causeID WHERE "

If txtReqID > 0 Then

s = s & "tblRequest.requestID = " & txtReqID

Else

s = s & "(false"

If Not chkAll Then

s = s & " OR ownerID = " & getSupporter()

Else

s = s & " OR status = 3" ' cTaken

End If

If chkFirst Then s = s & " OR status = 1" ' c1st

If chkSecond Then s = s & " OR status = 2" ' c2nd

If chkParked Then s = s & " OR status = 4" ' cParked

If chkClosed Then s = s & " OR status = 6" ' cClosed

s = s & " OR status = 5)" ' cReminder: Always include reminders

If txtFrom > 0 Then s = s & " AND startTime > " & CDbl(txtFrom)

If txtTo > 0 Then s = s & " AND startTime <= " & CDbl(txtTo)

End If

If lastRequest > 0 Then s = s & " OR tblRequest.requestID = " & lastRequest ' Always show latest request

Select Case cboOrderBy

Case 1

s = s & " ORDER BY requestID"

Case 2

s = s & " ORDER BY status"

Case 3

s = s & " ORDER BY owner.email "

Case 4

s = s & " ORDER BY priority"

Case 5

s = s & " ORDER BY startTime"

Case 6

s = s & " ORDER BY reminderTime"

Case 7

s = s & " ORDER BY tblcause.name"

Case 8

s = s & " ORDER BY sender.email "

Case 9

s = s & " ORDER BY subject"

End Select

If cboOrderBy = 5 Then

s = s & IIf(chkIncreasing, " DESC", " ") ' Ascending ages have descending startTime

Else

s = s & IIf(chkIncreasing, " ", " DESC")

End If

s = s & ";"

Me.subRequestList.Form.RecordSource = s

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboOrderBy\_AfterUpdate()

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkIncreasing\_AfterUpdate()

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdNewRequest\_Click()

DoCmd.OpenForm "frmRequest", , , , acFormAdd

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdOpenRequest\_Click()

If IsNull(subRequestList.Form.requestID) Then

Call MsgBox("There is no request in the list")

Else

DoCmd.OpenForm "frmRequest", , , "requestID = " & subRequestList.Form.requestID, acFormEdit

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdSeeSupporters\_Click()

DoCmd.OpenForm "frmSupporters"

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkAll\_AfterUpdate()

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkClosed\_AfterUpdate()

lastRequest = 0

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkFirst\_AfterUpdate()

lastRequest = 0

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_KeyDown(KeyCode As Integer, Shift As Integer)

If KeyCode = vbKeyF6 And Shift = 0 Then

oldFocus = ActiveControl.name

subRequestList.SetFocus ' Sets the selection area

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_Load()

Me.TimerInterval = 60000 ' One minute

If getTestMode() Then DoCmd.OpenForm "frmNow"

oldFocus = "chkFirst"

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_Timer()

If Not getTestMode() Then

s = "UPDATE tblRequest SET tblRequest.status = 5, tblRequest.reminderTime = Null WHERE tblRequest.reminderTime < Now();" ' Set reminders

CurrentDb.Execute s, dbFailOnError

subRequestList.Requery

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub txtFrom\_AfterUpdate()

lastRequest = 0

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkParked\_AfterUpdate()

lastRequest = 0

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkSecond\_AfterUpdate()

lastRequest = 0

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub txtReqID\_AfterUpdate()

lastRequest = 0

Call Search

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub txtTo\_AfterUpdate()

lastRequest = 0

Call Search

End Sub

# fsubRequestList

Option Compare Database

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_DblClick(Cancel As Integer)

DoCmd.OpenForm "frmRequest", , , "requestID = " & requestID, acFormEdit

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_KeyDown(KeyCode As Integer, Shift As Integer)

If KeyCode = vbKeyF6 And Shift = 0 Then

On Error Resume Next ' Maybe no old focus has been set

Parent(Parent.oldFocus).SetFocus

ElseIf KeyCode = vbKeyReturn And Shift = 0 Then

DoCmd.OpenForm "frmRequest", , , "requestID = " & requestID, acFormEdit

End If

End Sub

# frmSupporters

Option Compare Database

Option Explicit

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_AfterUpdate()

Dim res As Integer, st As Integer, s As String

If cboLine = c1st And DCount("empID", "qrySupporter", "line = 1 and not absent") = 0 Then ' No 1st liners left

Call MsgBox("Nobody is left to take care of 1st line", vbOKOnly)

End If

If chkAbsent And DCount("requestID", "tblRequest", "ownerID = " & empID & " AND status = 3") > 0 Then ' Requests taken by the supporter

If MsgBox("You handle some requests now. Transfer them to 1st line?", vbYesNo) = vbYes Then

st = c1st

ElseIf MsgBox("Transfer them to 2nd line?", vbYesNo) = vbYes Then

st = c2nd

End If

If st > 0 Then

s = "UPDATE tblRequest SET tblRequest.status = " & st & " , ownerID = NULL WHERE ownerID = " & empID & " AND status = 3;" ' Set reminders

CurrentDb.Execute s, dbFailOnError

End If

End If

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_KeyDown(KeyCode As Integer, Shift As Integer)

If KeyCode = vbKeyReturn And Shift = 0 Then Me.Recordset.Move (0)

End Sub

# frmEmail

Option Compare Database

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdOK\_Click()

DoCmd.Close

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub Form\_Load()

Const cm = 567 ' Twips

txtTo = parTo

txtFrom = parFrom

txtSubject = parSubject

txtBody = parBody

If mailRight = 0 Or maildown = 0 Then ' Initialize the position

mailRight = 3 \* cm

maildown = 3 \* cm

Else

mailRight = 9 \* 567 - mailRight ' Change between 3 and 6 cm positions. Otherwise it doesn't look like two messages when one is sent to the user and another to the supporter.

maildown = 9 \* 567 - maildown

End If

DoCmd.MoveSize mailRight, maildown

End Sub

# frmNow

Option Compare Database

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cboEmpID\_AfterUpdate()

Me.Recordset.MoveFirst

On Error Resume Next ' RequestList may not be open

Forms!frmRequestList.Search ' Recompute. Requery not enough since Current supporter is computed

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub chkTestMode\_AfterUpdate()

Me.Recordset.MoveFirst

testMode = chkTestMode

On Error Resume Next ' RequestList may not be open

Forms!frmRequestList.Search ' Recompute. Requery not enough since Current supporter is computed

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub cmdResetData\_Click()

CurrentDb.Execute "delete from tblHistory;", dbFailOnError

CurrentDb.Execute "delete from tblRequest;", dbFailOnError

CurrentDb.Execute "delete from tblSupporter;", dbFailOnError

CurrentDb.Execute "insert into tblSupporter select \* from tblSupporterInit where true;", dbFailOnError

CurrentDb.Execute "insert into tblRequest select \* from tblRequestInit where true;", dbFailOnError

CurrentDb.Execute "insert into tblHistory select \* from tblHistoryInit where true;", dbFailOnError

Me.cboEmpID.Requery ' Needed to allow choice from list

Me.cboEmpID = 7

Me.txtDateTime = #9/23/2005 8:04:10 PM#

Me.Recordset.MoveFirst

On Error Resume Next ' RequestList may not be open

Forms!frmRequestList.Search ' Recompute. Requery not enough since Current supporter is computed

End Sub

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Private Sub txtDateTime\_AfterUpdate()

Dim s As String

Me.Recordset.MoveFirst ' Save record. User may have tabbed to end of file.

s = "UPDATE tblRequest SET tblRequest.status = 5, tblRequest.reminderTime = Null WHERE tblRequest.reminderTime < " & CDbl(Me.dateTime) & ";" ' Set reminders

CurrentDb.Execute s, dbFailOnError

On Error Resume Next ' RequestList may not be open

Forms!frmRequestList.Search ' Requery not sufficient. The DB engine doesn't know that DLookup would find a new time.

End Sub

# Test data and test environment

During development I gradually developed the following test approach. I defined a table of requests with one request for each of the six possible request states. I stored them and their corresponding history records in the initialization tables, so that each test could start with a clean copy of these tables. The requests corresponded to something that realistically could be stored in real life, but they didn't pretend to show a realistic volume of requests.

I defined around 10 simple request causes. Furthermore I defined eight supporters and eight normal users. In two of the combo boxes, all employees appeared, including the supporters. To ease debugging, supporter names have (S) at the end so the tester can see whom he is selecting.

The reason I didn't define the test data early on was probably that I mentally grabbled with how the system would look in real-life situations with lots of data, and whether users could use it intuitively. The lesson was that this had to be two different sets of test data. The long one would not be suited for program testing.

When the program seemed to work okay in most cases, I started the systematic test. I defined test cases based on my own exam reply, which contained E/R model, task descriptions and design specification. I gradually defined test cases in the sequence shown in section 14. When needed, I modified the test data slightly to be able to test a special case in a simple way. During testing I kept the test case document open while I performed the tests. The test case list shows the bugs I detected at this stage.

At the end, I printed a list of all the programs and went through it looking for if-statements, loops, and other branches. For each of these I asked myself whether I already had a test case where these branches had been tested. If I couldn't identify a case, I noted it down as an additional test case. I had expected this to be a major task, but actually it barely took one hour and created 10 more things to test.

Whenever a test revealed a problem that I judged to be difficult to repair or dubious in some way, I recorded it as a defect. The defect list is in the next section.

In spite of careful testing, more errors turned up when a real user tried the system. These errors caused the program to crash. They are D13 to D16 below. The errors are also listed as test cases 1000 and on to make sure they are removed before release.

# Defects

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Defect description** | **Cure if known** | **Status** |
| D1 | Printing of requests mentioned in T2.2-2p is not supported in the design. |  | Pending |
| D2 | Finding the request when an external supplier replies, isn't supported in the design (T2.3-1) | Manually label the order with the requestID and/or the user's name. Allow requestID or user name as a search criterion. | RequestID provided. User name pending. |
| D3 | In the design of frmRequest, Cause was too far away from Subject. As a result the user didn't notice the easy way to fill both. | Rearrange fields. | Done |
| D4 | A request has many attributes and choosing what to show may be customer dependent. | Allow configuration of frmRequest, frmRequestList and history list. | Pending |
| D5 | Real mode: The current user (supporter) must be retrieved from somewhere (single sign-on). |  | Release 1 |
| D6 | Real mode: Email-sending must go through the real mail system. |  | Release 1 |
| D7 | Real mode: Incoming requests through Email must be captured. |  | Release 1 |
| D8 | Real mode: Multi-user system. Locking and sharing the database is needed. |  | Release 1 |
| D9 | Real mode: User provides more information or reopens a request on-line. |  | Future release |
| D10 | Would be convenient to see also the user's room number. Cf. D1. | Show in cbo list. Requires an additional attribute in tblEmployee. | Pending |
| D11 | *Causes* are hierarchical in tblCause but aren't shown as a hierarchical menu. Cause.descr not shown anywhere. |  | Pending |
| D12 | A DateTime Picker might be useful for the From- and To-dates in frmRequestList. Initial tests indicated two problems with it: It cannot show an empty date. It can only use rather large fonts. |  | Pending |
| D13 | If the request form is open and incompletely filled out, the system crashes when you try to open another request or create a new one. | Test whether frmRequest is open and tell the user to close it. | Pending |
| D14 | When creating a new request, the system crashes if you use the Save button before closing the Form. Saving requires that the record exists already. The record can only be created by the user through closing of the form. | Not known how to create the record from the program. Temporary solution: Remove the Save button. The user is puzzled but can close the form with Ctrl+F4 | Pending |
| D15 | During testing, a wrong sql-statement may be left as RecordSource in a fsub. It causes problems when the system is deployed and started. When the user has searched for something, the problem disappears. | Initialize the RecordSource at Open. | Done? |
| D16 | The system uses an old DAO library (Microsoft DAO 3.6). It is selected through Tools -> References. This reference may disappear when the application is moved to another computer that has a different setting. The result is a crash with a function-not-found error (or similar message). | Cure unknown. | Pending |

# Test cases

Test cases are numbered in bundles to ease insertion of new test cases where they belong. When the test consists of carrying out a user task, the individual steps are numbered according to the subtasks in the task description. The same task may be carried out several times in different variants. Each time it has a new test case ID. You may reset the data each time you start a task, but unless mentioned explicitly, you don't have to.

| **ID** | **Datamodel supported?** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 100 | **Request.** Read and edit all attributes and relationships. Use frmRequest. | (VBA error when setting Owner to blank. Corrected.) | ok |
| 101 | Check blank sender and source. Use frmRequest and tblRequest. Test setting to blank and undo. | Database checks only weak ref.integrity, but user interface checks for blanks at BeforeUpdate. (Setting sender to blank in UI caused VBA error. Corrected.) | ok |
| 102 | Check that cause and owner may be blank. Use frmRequest. Test setting to blank and undo. | (Setting to blank and then undo, crashed Access 2003. OK on Access 2000. The problem disappeared for unknown reasons.) | ok |
| 110 | **Cause.** Read and edit all attributes and relationships. Use frmRequest. | Hierarchy present, but not shown in UI. Descr present but not shown anywhere. | D11 |
| 120 | **Supporter.** Read and edit all attributes and relationships. Check referential integrity to employee. Use frmSupporter and tblSupporter. | Database checks ref. integrity. UI cannot attempt to violate it. | ok |
| 130 | **Employee.** Read and edit all attributes. (No relationships to be checked.) Use frmRequest. | (Columns too narrow in two combo boxes. Corrected.) | ok |
| 140 | **History.** Read all attributes and relationships. Edit should not be possible. Check referential integrity. | Database checks weak ref.integrity. UI cannot change history.  The following attributes are not visible through UI: reminderTime, subject, sender, currentPhone. Configuration options needed? | D4 |

| **ID** | **Work area 1: User** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 200 | **T1.1.** Request captured by mail. | Not implemented | D7 |
| 210 | **T1.2.** Follow up. | Not implemented | D9 |

| **ID** | **Work area 2: Hotline** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 300 | **T2.1. Request to hotline.** |  |  |
| 1 | Create user request. Omit user and phone. |  | ok |
| 1q | Set cause to Password. | Subject set automatically | ok |
| 1p | Close window. | Msg: Sender and subject must be set | ok |
| 2 | Set sender to Simpson. Close request F8. Close window. | Requestlist shows the closed request with an age of 1 min. | ok |
| 301 | **T2.1.** |  |  |
| 1a | Request created through email | Not implemented | D7 |
| 302 | **T2.1.** |  |  |
| 1b | Reminder. Open reminder request. |  |  |
| 2 | Write that the server is now ok. Close request F8. Close window. | System sends mail with the note and that the request is closed. | ok |
| 303 | **T2.1.** |  |  |
| 1c | User nete provides new information about request 4. Find the request. | Okay based on requestID, but if there were many requests and user couldn't remember the requestID, we would need a search on name | D2 |
| 3 | Add a note that she got a new door card yesterday. Leave request in in-tray. |  | ok |
| 304 | **T2.1.** |  |  |
| 1d | User wants to reopen request 26. Make sure closed requests are not shown. Then find the request. |  | ok |
| 4 | Add note "Needs a new password" and transfer to 2nd line. Give priority High and reminder in 2 hours. Close window. Say no to sending message. |  | ok |
| 4 | Check that it is in requestList. Open the request and check history. Check both views in both sequences. |  | ok |
| 305 | **T2.1.** |  |  |
| 1e | New information or new information provided online by user. | Not implemented | D9 |
| 306 | **T2.1.** |  |  |
| 1f | Request has been in 1st line for some time. Find the oldest. |  | ok |
| 4  4p | Transfer to someone who is present now and has the right expertise. Simpson and amc both have the expertise. Check that the supporter gets a warning message. | (The cbo didn't show the presence of the supporters. Corrected.) | ok |

| **310** | **T2.2. Request to 2nd line** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| start | Look at 2nd line requests and find the oldest (req 4). |  | ok |
| 1 | Check that the phone number is available for contacting the user. |  | ok |
| 2 | Check that the room number is available for walking to the user's location. | Not in the system. Would have to ask the user or look up the data. | D10 |
| 2p | Make list of rooms to visit. | Not supported. | D1 |
| 3 | Work on the request: Take it. Record a note to the user and send it, e.g. "Will take some time". Close the window. Check requestList. | (OK. But also sends mail to myself.  Repaired.) | ok |
| 4 | Open the request again and park it. Set a new reminder time. Close the window. | The user is informed of the new time. | ok |
| 311 | **T2.2.** |  |  |
| 5 | Open the reminder request (24). Assume you contact someone about it. Park it. Close the window. Say yes to setting the reminder time. | Msg: Set reminder time? | ok |
| 5p | Set the reminder to one week. Close the window. Say no to sending a message to the user. | Msg: Send message? | ok |
| 6  6p | Open request 24. Close request F8. Close window. Set a cause as suggested. Close window. Say yes to sending a message to the user. | Msg: Set a cause please.  Msg: Send message?  Message sent.  Request should still be in the list in spite of other closed messages not being shown. | ok |
| 312 | **T2.2.** |  |  |
| 7 | Open request 4. Transfer it to amc. Close window. | Mail to amc.  Request should still be in the list. | ok |

| **320** | **T2.3. Message from an external supplier** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 1 | Find the request bases on reqID or user name | See 303-1c | D2 |

| **330** | **T2.4. Change of role** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 1  2p | Open frmSupporter. Make knt absent. System warns and suggests to transfer them. Say No to 1st line, yes to 2nd line. | Request list shows that they are transferred to 2nd line | ok |
| 1 | Make kevin (yourself) absent. System should warn that no 1st liners are left. | (Bug found. Corrected.) | ok |

| **ID** | **Function test** | **Result** | **Correct?** |
| --- | --- | --- | --- |
|  | **frmRequestList** |  |  |
| 400 | Check all boxes and clear From/To. | Shows 7 requests. One closed (red), one reminder (yellow). | ok |
| 401 | **Search and OrderBy.** Uncheck all boxes one by one. Only the reminder should remain when all boxes are off. | Requests disappear correctly. | ok |
| 402 | Change current reporter to knt (use frmNow). | The request taken by knt appears. | ok |
| 403 | Check that the list can be ordered by any column, increasing. | (Four bugs found. Corrected.) | ok |
| 404 | Check that the list can be ordered by any column, decreasing. | (One bug found. Corrected.) | ok |
| 405 | Set From to 23-9-5 12:00. | Request 1 disappears. | ok |
| 406 | Set Last to 23-9-5 14:00. | Requests 24, 25, 26 disappear. | ok |
| 407 | Clear From. | Only requests 1, 2, 3, 4 appear. | ok |
| 408 | Set Request ID to 23, then 24. | No requests, then only 24 appears. | ok |
| 420 | **CreateReques**t. Then cancel with Esc. | Opens new request. Closes it without leaving anything in the list. | ok |
| 430 | **OpenRequest.** Click button. Close window with Ctrl+F4. | Should just open and close the request window. | ok |
| 431 | **SelectRequest.** Select with mouse, then Alt+R. Select with mouse, then Enter. Double click. | They all open the request window. | ok |
| 432 | **SeeSupporters.** Click button. Close frmSupporters with Ctrl+F4. |  | ok |
| 440 | **Function presentation.** Try all shortcuts. Try tab sequence. Try F6 to toggle between subform and main form. |  | ok |

|  | **Timer** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 500 | Reset data and include all requests in frmRequestList. Request 24 should be a reminder. Set current time to 23:04 (3 hours ahead). | Request 2 becomes a reminder. All Ages and Left are adjusted by 3 hours. | ok |
| 501 | Set current time to 24-09-05 14:04. | Request 1 and 3 should also become reminders. Knt who has taken request 3 and want email notification, should get a mail. | Knt doesn't get a mail.  **Pending** |

|  | **frmRequest** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 600 | **Create request** (from RequestList). | ReqID, Start, Source and Status set. Note shows "New note". Other fields empty. | ok |
| 601 | **SaveRequest - reqs 1 + 2.**  Set Sender to something. Close window with Ctrl+F4. | Msg: Sender and subject . . .  (Bad message. Improved.) | ok |
| 602 | Set Cause to something. Close window with x-icon. Ignore setting reminder. | (Bugs. Corrected.) | ok |
| 603 | **Create request**. Set subject. Close window. | Msg: Sender ... | ok |
| 604 | Set sender. Close request F8. Save F5. Say OK to setting cause. | Msg: Set cause. | ok |
| 605 | Set cause. Save F5. Cancel sending mail to user. | Msg: Send mail to user? | ok |
| 606 | Set note. Save F5. Say yes to send mail to user. Check history line.  Check request list. | Msg: Send mail to user?  One history line.present.  (After F5 the request didn't behave like a fully created request. Corrected.) | ok |
| 607 | **Create request.**  Set all fields, also Note and Email.  Tab to end to check that tab cycles within the record. Try PgDn and PgUp.  Save F5. Check history. Close window. Check request list. | Mail to user.  One history line.  (Several bugs: Cycling was allowed. Saving didn't set the filter correctly. Corrected.) | ok |
| 610 | **Existing request.** **Reset data.**  Open request 24. Tab to end to check that tab cycles within the record. Try PgDn and PgUp. |  | ok |
| 611 | **SaveRequest - req 1.**  Delete sender. Save F5. Esc to reset sender. | Msg: Sender cannot be blank. | ok |
| 612 | Delete subject. Save F5. Esc to reset subject. | Msg: Sender and subject must be defined. | ok |
| 613 | **SaveRequest - reqs 2 + 3 + 4 + 6 + 7.**  Close F8. Save F5. Say OK to setting cause.  Set cause. Save F5. Say Yes to send mail to user. | Msg: Set a cause ...  New history line.  Mail to user with standard text. | ok |
| 614 | Set state to 1st. Save F5. Cancel setting reminder.  Set note. Check Email. Close F8. Save F5. | Msg: Set reminder?  New history line.  No question. Note sent as mail to user. | ok |
| 615 | **SaveRequest - req 5.** Set state to 1st. Set reminder. Save F5. Say yes to sending mail. | Msg: Send mail?  Mail to user with standard text. | ok |
| 616 | Set reminder to something else.  Set note. Check Email. Save F5. | No question. Note sent as mail to user. | ok |
| 617 | **SaveRequest - req 8.** Take request F7. Save F5. | No message | ok |
| 618 | Set owner to amc. Save F5. | Message to amc.  (Message text corrected.) | ok |
| 630 | **Cancel.** Change all fields. Esc to reset data. |  | ok |
| 631 | **History.** Toggle Text View and Increasing. | There are several history records. | ok |
| 632 | **Email.** Set note. Set Email (if not set). Save F5. | Message to sender. No questions. | ok |
| 640 | **Data entry.** Set owner. | Status becomes Taken | ok |
| 641 | Set status to 1st. | Owner becomes blank. | ok |
| 642 | Take request F7 and Close F8: Tested several times above. |  | -- |
| 643 | Cause. Tested several times above. |  | -- |
| 644 | Reminder. Enter directly into field. Select None from list. Select something else from list. |  | ok |
| 645 | Sender set several times above.  Set sender name. | (Autocomplete wasn't set. Corrected.) | ok |
| 646 | Phone. Overwrite.  Then select a new sender. | New phone stored.  New sender's phone shown. | ok |
| 647 | Copy history. Paste into Notepad or something else. | OK for text view.  (List view cannot copy the whole list - OK) | ok |
| 648 | Blank these fields (should be allowed): Owner, Priority, Reminder, Phone, Cause, Subject, Note. |  | ok |
| 649 | Blank these fields (should not be allowed): ReqID, Status, Source, Start, Sender, SenderName. | (Status gave strange messages. Corrected.)  Error messages, blocked or default set. | ok |
| 660 | **Function presentation.** Try all shortcuts. Try tab sequence. |  | ok |

|  | **frmSupporter** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 700 | Tested in 330 (T2.4. Change of role). |  | -- |

|  | **Real Mode** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 800 | **Test mode.** Reset data. Set test mode off (in Now). Close the Hotline system. Open it again. | frmNow doesn't appear.  After a minute, all ages are updated to several years. A bit later all requests are set in reminder state. | ok |
| 801 | **Mail sending.** Open a request. Set note and Email. Save F5. | Real email should be sent. | D6 |

|  | **Glass box test - Cases not tested above** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 900 | **basCommon.** Solid(null). Only called with Status as parameter, but status is now enforced to non-null. Tested with debugger. |  | ok |
| 910 | **frmRequest.** cbReminder\_AfterUpdate:  Set Reminder to None through the dropdown. |  | ok |
| 911 | Form\_AfterUpdate. Switch to Text View. Change the request and Save F5. | History in text form is updated. | ok |
| 912 | Form\_AfterUpdate. Check that taking a request doesn't generate a message to yourself. Make amc the current supporter (wants email notification). Open request 2 and take it. | No message to amc. | ok |
| 913 | Test that updates work okay when frmRequestList is closed. Open request 2 and Close frmRequestList. Set owner to jul (wants email notification). Save F5. | Message sent to jul. History list updated. | ok |
| 920 | **frmNow.** Test that update works okay when frmRequestList closed. Close frmRequestList. Change Now. Enter. Change empID. Enter. Change Testmode. Enter. Reset data. Enter. |  | ok |
| 930 | **frmError.** Reset data. Open frmRequestList. Create new request. Set Sender and close form with Ctrl+F4. Don't set subject, but click the x standard close button. Finish with Esc. | Error message twice. You are allowed to correct the request.  Esc closes and deletes the new request. | ok |
| 940 | **frmrequestList.** Search: Test display of negative ages: Set time to 23-08-05. | Ages in frmRequestList become negative (around -30 days) | ok |
| 941 | cmdOpenRequest: Test that list can be empty: Set ReqID to a non-existing requestID. Click OpenRequest. | Error message. | ok |
| 950 | **fsubRequestList.** Form\_KeyDown: Test that F6 works when F6 has not been used in main form: Close frmRequestList and open it again. Set focus in fsubRequestList. Use F6. | Switches focus to main form. | ok |

| **ID** | **Errors detected after release** | **Result** | **Correct?** |
| --- | --- | --- | --- |
| 1000 | Open frmRequest and make an incomplete change. Then try all functions in frmRequestlist. | The system crashes if you try to open another request or create a new one. Cure: Test whether frmRequest is open and tell the user to close it. | pending |
| 1001 | Create a new request and then use the Save button before closing the Form. | The system crashes because saving the record requires an existing record. Closing the form is necessary. | pending.  Temporary fix: The Save button is removed. |
| 1002 | During testing, a wrong sql-statement may be left as RecordSource in a fsub and later shipped with the new release. | It causes problems when the system is deployed and started. When the user has searched for something, the problem disappears. Cure: Initialize the RecordSource at Open. | pending |
| 1003 | The system uses an old DAO library (Microsoft DAO 3.6). It is selected through Tools -> References. This reference may disappear when the application is moved to another computer that has a different setting. | The system crashes with a function-not-found error (or similar message). | pending |