Chapter 6

Implementation

6.1 Introduction

The previous chapter was about the analysis and design of the proposed system. This chapter describes implementation of said design. Each module which is described in the architectural design will be implemented separately.

6.2 Upload Data

In the Inward data DVD the branch wise data is in separate folders and in these folders a data file and the cheque images are in another two folders. Cheque images are in Tag Image File Format – TIFF. In the upload process following will done in sequentially.

- Get Branch Codes into a hidden list
- Get the names of the images of the first branch into another hidden list. Name of the images are form with the UI No.s.
- Open the data file of the firsts Branch Code of the list
- Insert the first data line of the data file (UI No., Account No., Cheque No, Amount, Presenting Bank/Branch codes etc included)
- Search for the same UI No. in image name list
- If the name found update the record by adding the cheque image. In order to add the add image following steps has to follow.
    - convert the TIFF image in the DVD in to a binary stream
    - Update the database with this binary steam (to the same record)

  The reason for converting the image into binary stream the picture of TIFF can not be stored in a database directly as a picture.
- Follow the steps said above until the finishes the lines of data file.
- Once it finishes open the next branch's data file and clear the hidden list of names of images
- Get the names of the images of the current branch into the hidden list

- Repeat the steps from onward
- Do these steps until all the branches finishes

For the screen shots refer Appendix H.

The representation of above steps in pseudo codes-

"Get Branch list into BrList

While BrList Finishes

Open_Data_File Branch 1

While Data_File1 finishes

Insert Data (UI No, Account No,Cheque No etc)Into Database

Search for this UI No in Image List

If found the image

Convert image to Binary Stream

Update the record with Binary Stream

Else

Delete All Branch Data

End if

Move To Next Data Line

End While

Move to Next Branch

End While"

The Visual Basic Code segment for convert TIFF image to Binary Stream and Update the recode is shown in Figure 6.1.

```
Public Function SavePictureToDB(Rs As ADODB.Recordset, sFilename As String)
  Dim strStream As ADODB.Stream
  Set strStream = New ADODB.Stream
  strStream.Type = adTypeBinary
  strStream.Open
  strStream.LoadFromFile sFilename
  With Rs
        .Fields("CHQ_IMAGE").Value = strStream.Read
      .Update
  End With

  SavePictureToDB = True
procExitSub:
  Exit Function
procNoPicture:
  SavePictureToDB = False
  GoTo procExitSub
End Function
```

Figure 6.1 : Visual Basic Code segment for convert TIFF image to Binary Stream

6.3 Allocation

Allocation is consisted with two segments

- Set the Allocation Matrix – How allocation distribute among users
- Allocation as per the matrix

As per the allocation is concern the process is to update records in "INWARD_CLEARING" table's "ASSIGNED_USER" field. Setting the Allocation Matrix is build the "Where clause" of the "Update" statement. The fields and their values of the "Where clause" and the Assign User are got by the "ALLOCATION_MATRIX" table.

The Update statement is as follows:

A- *ALLOCATION_MATRIX*

"Update INWARD_CLEARING set Assigned_User=A.USER_ID where
A.FIELD_NAME = A.Value"

The representation of above in pseudo codes-

"Record Set1=Select * from ALLOCATION_MATRIX

While Record Set1 finishes

Update INWARD_CLEARING Update INWARD_CLEARING set

Assigned_User=A.USER_ID where A.FIELD_NAME = A.Value

End While"

6.4 Scrutinizing

This is the process of view cheque images along with data to the assigned users. In order to view images it is need a view controller. Here Microsoft Office Document Imaging – MODI is used for this. As a particular user navigate to scrutinizing screen followings should be happen.

- Go to View Criteria screen and setting of this screen makes the criteria to view cheques
- Get the all the records of INWARD_CLEARING table for the user and as per the said criteria
- View the first cheque along with relevant data and the signature(s) from the SVS.
- Allow user to mark the cheque.

The representation of above steps in pseudo codes-

"Build the where clause

Get the record set from INWARD_CLEARING as per where clause

The binary stream of the image of first record save the in a folder

set MODI.filename= "path/ImageName"

Show the other data

Show the signature(s)"

The Visual Basic Code segment for convert Binary Stream to TIFF image and show the image is shown in Figure 6.2.

```
Public Function LoadPictureFromDB(db As ADODB.Connection, Rs As ADODB.Recordset, MDV As MiDocView, thisForm
As Form, AbPos As Integer)
On Error GoTo procNoPicture
Dim strStream1 As ADODB.Stream

  Set strStream1 = New ADODB.Stream
  strStream1.Type = adTypeBinary
  strStream1.Open

  strStream1.Write Rs.Fields("CHQ_Image").Value
  Dim CC As String, aa As String
  CC = "C:\rr" & ".tiff"
  strStream1.SaveToFile CC, adSaveCreateOverWrite
  FileCopy "C:\rr" & ".tiff", "C:\98Old\rr" & ".tiff"
  MDV.FileName = CC
  MDV.PageNum = 1
  LoadPictureFromDB = True

procExitFunction:
  Exit Function
procNoPicture:
```

Figure 6.2 : Visual Basic Code segment for convert Binary Stream to TIFF image and show the image

6.5 Create Files

Three data file to be created in the proposed system.

1. Corrected File to be sent to Core Banking system
2. The Outward return file to be burn to a CD
3. The technical return file to be sent to Core Banking system

Basic steps are as follows:

- Get data selected form INWARD_CLEARING table.
- Make the text file in agreed format.

6.6 View Cheques by Customers

For this module web base is used. Relevant steps are as follows.

- Customers login to system
- Then display list of own cheques numbers presented in the day
- By selecting each cheque number view the relevant image of cheque

The PHP Code segment for view cheques for the relevant customer is shown in Figure 6.3. The session variable "maccno" and "mchqno" are coming from the previous screen.

```
<?php
session_start();
$maccno=$_SESSION["maccno"];
$mchqno=$_SESSION['mchqno'];

$link = mssql_connect("10.1.2.30", "sa", "#imago123");
mssql_select_db("Project",$link);

$sql = "select CHQ_IMAGE from INWARD_CLEARING where
            ACCOUNT_NUMBER='$maccno' and CHEQUE_NO='$mchqno'";

$row=mssql_fetch_array($rs);
$data=$row['CHQ_IMAGE'];

    $result = mssql_query("$sql",$link);
    $data=mssql_result($result,0,'CHQ_IMAGE');

    echo $data;
    mssql_close($link);
    exit();
?>
```


Figure 6.3 : PHP Code segment for view cheques for the relevant customer

6.7 Handle Message Format

As this module is depending on the Core Banking System vendor it has not been completed.

This module is needed when it is come to "Correcting Data" validation. For the prototype it was arranged an internal table and do the validation.

6.8 Correcting Data

Same as scrutinizing view the cheques. Then correct the Account No. or/and Cheque No. with validating Core Banking System. If both Account No. and Cheque No. changed authorization is required.

The Visual Basic code segment of validate Account No. is shown in Figure 6.4.

```
Private Sub ValAcc()

If txtAccNo <> "" And Val(txtAccNo) <> 0 And Len(Trim(txtAccNo)) = 10 Then 'txtAS4 = 1
Then
  InitialVal
  Screen.MousePointer = 11
  C = ChkD(frmACUp)
  Acc = Mid(lblBr, 1, 3) & "1" & Mid(txtAccNo, 8, 3) & C & Mid(txtAccNo, 1, 7)


    Call OpenRec("SELECT * FROM Accounts where AccNo='" & txtAccNo & "' ", db, Rs1)

    If Rs1.EOF Then
      lblCust.ForeColor = &HFF&
      lblCust.Caption = "Invalid Acc"
      lblChq.ForeColor = &HFF&
      lblChq.Caption = "Invalid Chq"

      'ValChq
    Else
      lblCust.ForeColor = &H80000002
      lblCust.Caption = Rs1(1)

      If Len(txtChq) = 6 Then
        ValChq
      End If
    End If
  Screen.MousePointer = 1
End If
End Sub
```


Figure 6.4 : Visual Basic code segment of validate Account No.

6.9 Following Signature Rules

In the SVS there are rules are defined in terms of amount. When it is show the relevant signature(s) at the scrutinizing these rules must be followed. And if there is more than one signature to be showed they have to be merged in to one image.

The Visual Basic code segment of merge signatures is shown in Figure 6.5.

```
While Rs1.EOF = False

  Set strStream1 = New ADODB.Stream
  strStream1.Type = adTypeBinary
  strStream1.Open

  strStream1.Write Rs1.Fields("SIGNATURE").Value

  Dim CC As String

  If Dir("C:\98Old\qq.tiff") <> "" Then
    Kill ("C:\98Old\qq.tiff")
  End If
  CC = "C:\" & i & ".tiff"

  strStream1.SaveToFile CC, adSaveCreateOverWrite

  FileCopy CC, "C:\98Old\INV\" & i & ".tiff"

  Set miDocSrc = New MODIctl.Document
  Set miDocSrc1 = New MODIctl.Document
  miDocSrc.Create "C:\98Old\INV\" & i & ".tiff"

  With miDocDest.Images
     .Add miDocSrc.Images(0), Nothing
  End With

  Set miDocSrc = Nothing
  If Dir("C:\98Old\INV\" & i & ".tiff") <> "" Then
     Kill ("C:\98Old\INV\" & i & ".tiff")
  End If

  i = i + 1
Rs1.MoveNext
Wend
```

Figure 6.5 : Visual Basic code segment of merge signatures

6.10 Burn Return Data

For burning CD purpose the "NeroCMD.exe" application is used. A batch file is created to call NeroCMD.exe and burn the file.

6.11 Handle Additional Days

When burn the return data it can be some branches get additional day for submitting return data. It should be tagged them as additional day and record how many days take as additional.

6.12 Parameters

Here it is defined the parameters of the system. Under this following were implemented.

- CD Rom drive : where Inward data DVD uploaded from
- CD Writer Drive : where Outward Return data burn to CD

- Backup Drive : where backup is stored
- Signature Server : IP address of the SVS server
- Bank Host : IP address of the Core Banking Server
- Change Password Cycle (in days)

6.13 Calendar

In order to keep a track of holidays this module is used. As far as the system is concern the processing day is validated. To keep a track of status of a day a string of 1s and 0s is stored in a "CLEARING_CALENDAR" table.

The Visual Basic code segment of changing the day string as day status is change in the screen is show in Figure 6.6.

```
Private Sub Check_Click(Index As Integer)
Dim i As Integer, DD As Integer, EE As Integer
For i = 0 To 41
  If Check(i).Caption <> "" Then Exit For
Next
If txtStat = 1 Then

  DD = Check.Item(Index).Caption
  If Check.Item(DD).Value = 0 Then
    Check.Item(DD).BackColor = &HFF&
    txtCBit = Mid(txtCBit, 1, DD - 1) & 0 & Mid(txtCBit, DD + 1, Len(txtCBit))
  ElseIf Check.Item(DD).Value = 1 Then
    Check.Item(DD).BackColor = vbWhite
    txtCBit = Mid(txtCBit, 1, DD - 1) & 1 & Mid(txtCBit, DD + 1, Len(txtCBit))
  End If
ElseIf txtStat = 2 Then
  DD = Check.Item(Index).Caption + i - 1
  EE = Check.Item(Index).Caption
  If Check.Item(DD).Value = 0 Then
    Check.Item(DD).BackColor = &HFF&
    txtCBit = Mid(txtCBit, 1, EE - 1) & 0 & Mid(txtCBit, EE + 1, Len(txtCBit))
  ElseIf Check.Item(DD).Value = 1 Then
    Check.Item(DD).BackColor = vbWhite
    txtCBit = Mid(txtCBit, 1, EE - 1) & 1 & Mid(txtCBit, EE + 1, Len(txtCBit))
  End If
End If
End Sub
```


Figure 6.6 : Visual Basic code segment of changing the day string

6.14 Modules Implemented

For the sake of facilitate to maintenance it is necessary to declare the file names of each modules been implemented. There is mixture of ".frm", "bas" and ".php" file names

available as far as the CICPS is concern. The table 6.1 shows the said modules and corresponding file names.

Module	File Name(s)
Upload Data	frmLoadCD.frm
Allocation	frmAllo1.frm/ frmAllo2.frm
Scrutinizing	frmView2.frm
Create Files	frmBurn2.frm/ AS400.bas
View Cheques by Customers	login_new.php/ chq_select.php/ sig2.php
Correcting Data	frmACUp.frm
Following Signature Rules	SVS.bas/ frmView2.frm
Parameters	frmSysPara.frm
Calendar	frmCAL.frm

Table 6.1: Modules and File Names

6.15 System Dependability

All of us are familiar with the problem of computer system failure. For no obvious reason, computer systems sometimes crash and fail to deliver the services that have been requested [20]. The achievement of some of principals of dependability as follows:

6.15.1 Availability

The system will be tested for delivering service when the users need it at different scenarios.

6.15.2 Reliability

The system has been tested with the original specifications to check whether it deliver the desired functionality over a given period of time.

6.15.3 Security

It is a critical issue of securing a system by intrusions. Followings were considered when the system is implemented.

---

[20] Ian Sommerville – Software Engineering - Seventh Edition Page 69

- **User Password encryption**

It was used 64 bit conversion of password and an encryption key which is known to only two parties.

- **User Password Change Cycle**

User password change cycle was implemented and the cycle (no. of days to change) was implemented as a parameter. The users are NOT allowed used their last six passwords as their new password.

- **User Password Length**

The minimum length of password allowed to use is six.

- **User Levels**

User levels were implemented considering users' role in the system. By means of that restrict accessing system functions which are not assign to a particular user. Menu Items were restricted with regard to this.

- **Databases**

The passwords of databases were stored in the database itself and they were also encrypted. The IP address of servers of databases (i.e Core Banking database, SVS database) is also stored in the database. Hence achieve the avoiding the revelling of actual servers that the users are working on. It makes ease the maintainability too.

- **Sessions Control**

In the web base implementation, it is essential to implement proper security, as it is used by the out side personal. The "php.ini" file that is in the Apache web server configures the amount of time a session exits. The expiring a session the user will be re directed into login page automatically.

6.15.4 Maintainability

This principal is achieved by maintaining reusable programs in modules. Hence achieve the ease of program changes. Implementing system parameters too pave the way to ease the maintainability.

E.g.:- In a case of fail the Core Banking Server the Disaster Recovery Centre has mirror server of that. By simple changing the Core Banking Server into mirror server's IP address the CICPS can deliver the continuous service.

6.15.5 Error Tolerance

Here it is considered the design of the system to user input error are avoided and tolerated. When user errors occur, the system should, as far as possible, detect these errors and either fixes them automatically or request the user to re-input their data [21].

- Check Digit

When the Account No. enters the predefined check digit is validated.

- Length of Account No. and Cheque No.

As the said lengths are pre defined, they were implemented in the system too.

- Avoiding entering characters in Numeric Fields

When it is allowed only numeric values, the other characters were detected and avoided.

6.16 Summary

This chapter describes implementation of proposed system. It included pseudo codes and code segments when necessary. It also discussed the dependability of the system. The next chapter is discussed about evaluation of the system.

[21] Ian Sommerville – Software Engineering - Seventh Edition Page 71