3. Diseño e implementación del sistema

En este capitulo se describirá la arquitectura y la implementación de este sistema.

3.1. Análisis de las necesidades

Para poder generar esta interfaz, se requiere de un lenguaje de programación que pueda soportar llamadas y creación de Servicios Web, conexiones a bases de datos para almacenar las reservaciones que se reciben de los operadores.

Los requisitos de hardware que se necesitan para este sistema es una computadora personal corriendo Microsoft Windows XP.

La programación se realizara basándome en productos Microsoft Visual Studio.Net, y Java. Para el almacenamiento de datos, se utilizara MySQL, debido a la facilidad que proporcionan para su instalación, configuración y uso, y a que la licencia es de uso público.

Lo primero en el desarrollo de este sistema, será generar un sistema simulador de las cuatro diferentes tecnologías (XML, Flat File, Email, y Portal). Para esto, me basare en el lenguaje Visual Basic.Net. Este simulador generar los mismos requisitos que un Portal u Operador utiliza. El simulador necesitara enviar notificaciones de reservaciones vía Email,

y para el propósito de esta tesis, se creo una cuenta en Gmail para almacenar las reservaciones simuladas (tesis.reservas@gmail.com)

3.2. Arquitectura del Sistema

Para poder presentar este sistema, se requiere básicamente de tres módulos, el simulador, y el decodificador, y el generador.

3.2.1. Simulador

Se requiere de crear un simulador de reservaciones en las cuatro diferentes tecnologías para poder demostrar el ciclo completo del proceso de la decodificación sin utilizar información privada. La información de las reservaciones en la base de datos contiene nombres ficticios.

Este modulo de simulación tiene las siguientes funciones:

- Generación de reservaciones XML en formato OTA
    - Envío de reservación XML generada en formato OTA a un Servicio Web para su decodificación.
- Generación de reservaciones Flat File basándose en formato TMTC.
- Generación de reservaciones Email basándose en el formato Cheap Caribbean.

- Generación de reservaciones en Portal Web en formato Thomson

El funcionamiento de este sistema comienza cuando seleccionamos una de las funciones de generación de reservaciones. Al comenzar esta tarea, el sistema se conecta a la base de datos local para recuperar información de las reservaciones, previamente insertadas manualmente, y poder generar los diferentes formatos.

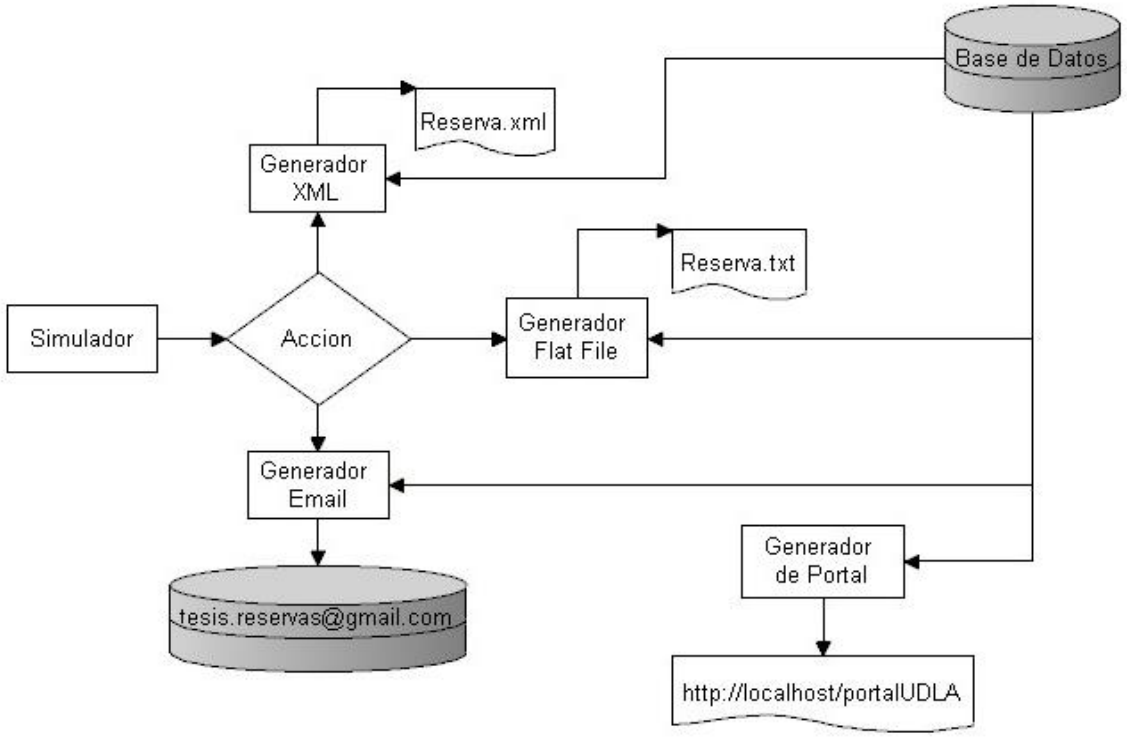
En el caso de los archivos XML, estos se generan, y se depositan en un directorio para que posteriormente se envíen al Servidor Web de decodificación.

Las reservaciones Flat Files, se depositan en un directorio para estas ser procesadas por el decodificador.

Los Emails son generados y enviados a tesis.reservas@gmail.com.

El portal esta hospedado localmente en http://localhost/portalUDLA, y recuperara información de las reservaciones en el momento en que un cliente valide su entrada.

A continuación se muestra el diagrama de acciones generales del simulador.



Gráfica 3.2.1. Diagrama de acciones generales para el simulador

3.2.2. Decodificador

El sistema principal es el decodificador. El cual consta de cuatro módulos independientes donde comparten una misma base de datos para el almacenamiento. A continuación se describirán estos módulos:

- ***Servicio Web***

Este modulo es un Servicio Web almacenado en http://localhost donde la operación principal es *processOTA* esperando como parámetro único un archivo XML basado en las especificaciones OTA_*HotelResNotifRQ*. El objetivo de este servicio es decodificar la reservación en este formato, para posteriormente insertarla en una tabla de transferencias en la base de datos MySQL. Este modulo esta basado en la plataforma Visual Studio.Net, utilizando como lenguaje a Visual Basic.Net. Para el funcionamiento de este modulo, se requiere la configuración de Servidor Web IIS (Internet Information Server) con las extensiones ASP.Net para su propia ejecución.

- ***Flat File***

El objetivo principal de este modulo es la decodificación de los archivos generados por TMTC (The Mark Travel Corporation). Para esta decodificación, el modulo busca archivos en el directorio de recepción (/incoming) basándose en el día actual (el formato para el nombre de estos archivos es <código hotel><DDMMMYY>.txt, y en caso de existir algún archivo, este comienza con la decodificación, para posteriormente, insertarlo en la tabla de transferencias de la base de datos. Este modulo se desarrollo en Visual Basic.Net, y la base de datos MySQL. Este modulo deberá ejecutarse una vez al día, ya que los operadores transmiten en la madrugada todas las reservaciones almacenadas durante el día anterior en esa transmisión diaria. Para nuestro propósito, esta aplicación se ejecutara manualmente.

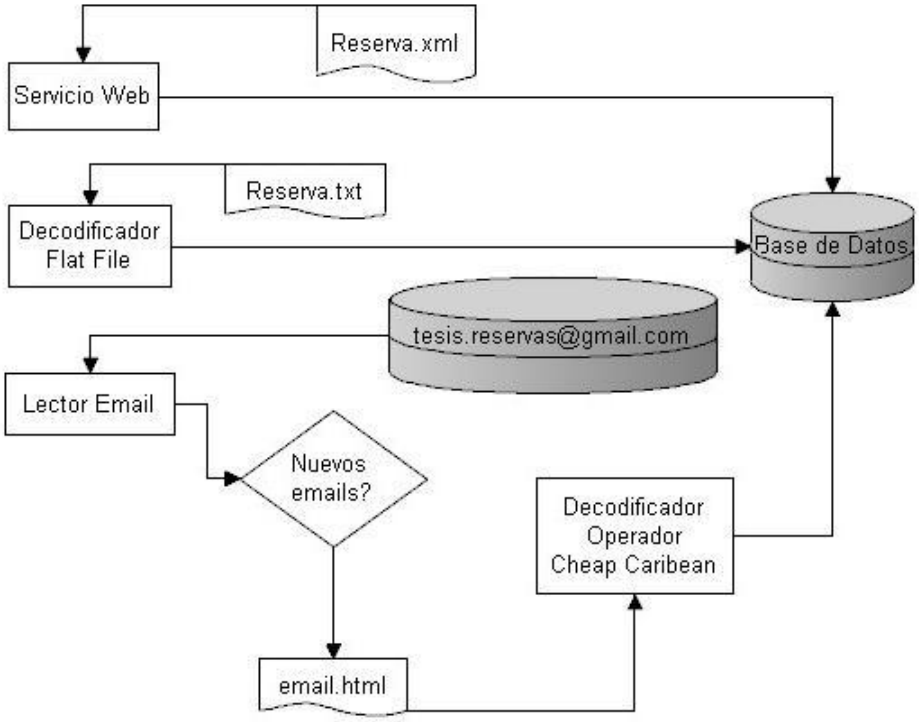
- Email

Este método consta de dos procesos principalmente: La función del primer proceso es conectarse a la cuenta de email tesis.reservas@gmail.com, y verificar si existen nuevos correos provenientes de nuestro proveedor de reservaciones Cheap Caribbean (en este caso, se utiliza la misma cuenta utilizada por nuestro simulador – tesis.reservas@gmail.com). En caso de existir nuevos correos, estos los baja a archivos HTML a directorios locales, para que el segundo proceso, que consta de la decodificación del archivo HTML, basándose principalmente en la búsqueda de campos dentro de la reservación HTML, pueda decodificarlos e insertarlos en la tabla de transferencias de la base de datos. Este segundo proceso, mueve los archivos procesados a un subdirectorio, para que estos no se procesen nuevamente. Para este método se implemento un sistema e alertas, que consta en la validación de la información que siempre transmite este operador, para que en caso de que cambie el formato de estos emails, notifique al desarrollador de estos cambios vía email, y aplique las nuevas reglas. El desarrollo esta basado en Java.

- ***Portal Web***

Este modulo tiene dos funciones principales, donde la primera es entrar al portal de Thomson en http://localhost/portalUDLA con el usuario y password del hotel, y primero saber el total de las reservaciones pendientes, para entonces hacer la segunda tarea que consta de hacer un ciclo de cada una de ellas, para recuperar los detalles. Para esta recuperación nos basamos en la búsqueda de etiquetas <HTML> para la recuperación de los detalles, tales como los nombres de los pasajeros, fechas de llegadas, salidas, tipo de habitación, y tipo de tarifa, entre otras. Al igual que en el sistema de emails, también se han implementado alertas al buscar información, para en caso de existir algún cambio en la presentación de información, envíe un email al desarrollador informando que ha existido un error, y pueda tomar las medidas pertinentes. La información recuperada será insertada en la tabla de transferencias de la base de datos, para que el trigger verifique su existencia y pueda saber si insertarla o no. El desarrollo de este modulo será en Visual Basic.Net, utilizando la base de datos compartida por los otros métodos en MySQL.

A continuación se muestra el diagrama general de las funciones de la decodificación.



Gráfica 3.2.2. Diagrama general de funciones para el decodificador

3.2.3. **Generador de reservaciones XML**

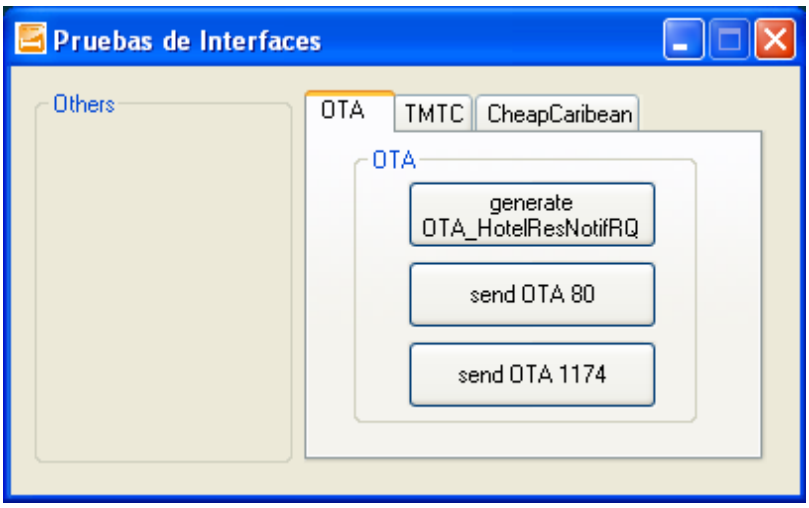
Para la generación de reservaciones XML en formato OTA se utilizara el simulador.

3.3. Implementación del sistema

En esta sección se detallaran las funciones principales de cada uno de los cuatro módulos, así como un ejemplo funciones que decodifican información.

3.3.1. Simulador

Comenzaremos con el algunos ejemplos de las principales funciones del simulador, nos enfocaremos únicamente a la generación de OTA XML y del portal, ya que nuestro objetivo principal es la decodificación de la información, el simulador es únicamente de soporte para la demostración de este sistema.



Gráfica 3.3.1a Pantalla principal del simulador

La simulación OTA XML requiere de conectarse a la base de datos general para generar el archivo XML, para después enviarlo al puerto 80, que es donde tenemos el Web Service esperando este tipo de formatos.

```
Public Function OTA_HotelResNotifRQ(ByVal xStatus As String) As String

        Dim vRes As String = ""
        Dim vNow As String = Now.ToString("yyyy-MM-ddTHH:mm:ss-05:00")
        Dim vEchoToken As String = RandomString(9, False)

…

        Try
            vConnection = openDB()

            If (xStatus.Equals("Commit")) Then
                ' Nuevas reservaciones
                vQuery = "Select " _
                       & "in_hotel, in_locator, in_reserva, in_charter, " _
                       & "in_room, in_rate, in_comments, date_format(in_arrival,'%Y-%m-%d'),
date_format(in_depart,'%Y-%m-%d'), " _
                       & "in_arrival_carrier, in_arrival_flight, in_arrival_time, " _
                       & "in_depart_carrier, in_depart_flight, in_depart_time, " _
                       & "in_source, in_adult, in_children, in_infant, " _
                       & "date_format(in_inp_d,'%Y-%m-%dT%H:%i:%s-05:00'),
date_format(in_mod_d,'%Y-%m-%dT%H:%i:%s-05:00'), date_format(in_cxl_d,'%Y-%m-%dT%H:%i:%s-
05:00') " _
                       & "from in_reserva " _
                       & "where in_mod_d is null and in_cxl_d is null and " _
                       & "in_charter = 'OTAXML'"

            ElseIf (xStatus.Equals("Modify")) Then

…

            vCommand = New MySqlCommand(vQuery, vConnection)
            vReader = vCommand.ExecuteReader()

            vRes = "<?xml version='1.0' encoding='utf-8'?>"

            ' OTA Header
            vRes = vRes + "<OTA_HotelResNotifRQ
xmlns='http://www.opentravel.org/OTA/2003/05' " _
                        & "xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance' " _
                        & "xsi:schemaLocation='http://www.opentravel.org/OTA/2003/05
OTA_HotelResNotifRQ.xsd' " _
                        & "Version='1.003' EchoToken='" + vEchoToken + "' ResStatus='" +
xStatus + "' " _
                        & "TimeStamp='" + vNow + "'>"

            While vReader.Read()
                vHotel = vReader(0).ToString()
                vLocator = vReader(1).ToString()
                If xStatus.Equals("Commit") Then
                    vRes = vRes + "<HotelReservation CreateDateTime='" + vInpD + "'
CreatorID='" + vLocator + "'>"
                ElseIf xStatus.Equals("Modify") Then
                    vRes = vRes + "<HotelReservation CreateDateTime='" + vInpD + "'
CreatorID='" + vLocator + "' LastModifyDateTime='" + vModD + "'>"
                ElseIf xStatus.Equals("Cancel") Then
                    vRes = vRes + "<HotelReservation CreateDateTime='" + vInpD + "'
CreatorID='" + vLocator + "' LastModifyDateTime='" + vCxlD + "'>"
```

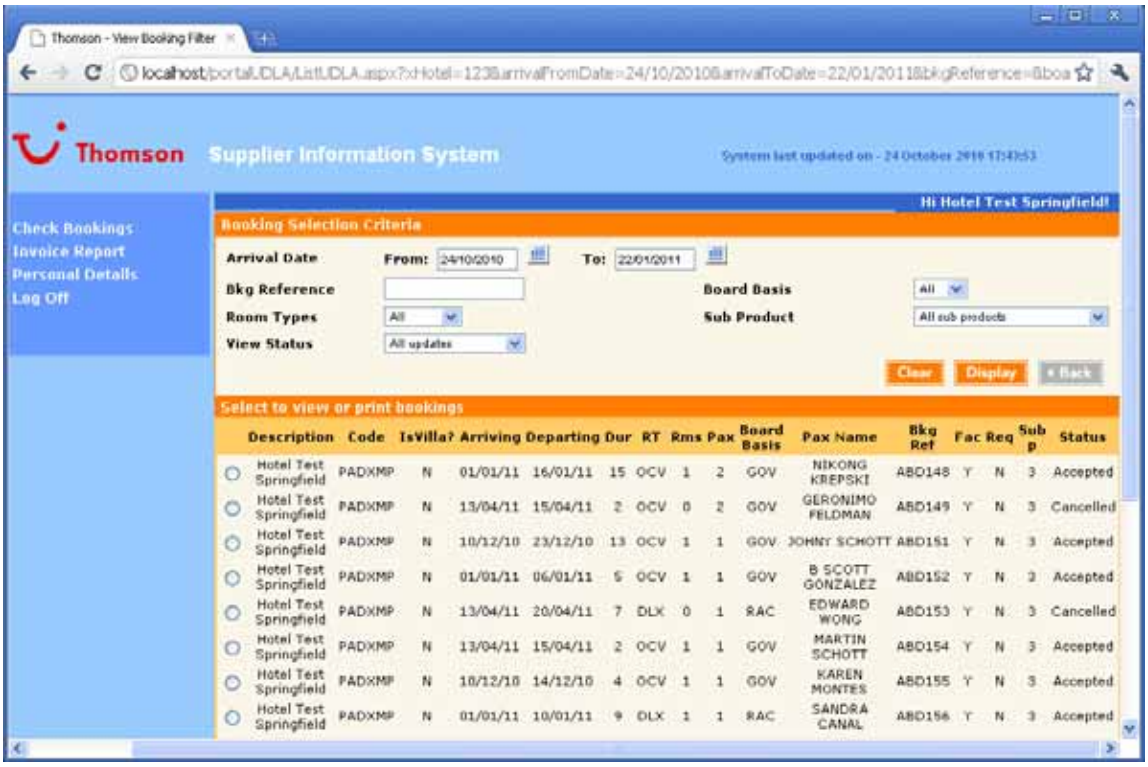
```
                End If

                vRes = vRes + " <UniqueID Type='14' ID='" + vReserva + "' />"
…
                vRes = vRes + "</HotelReservation>"
            End While

            vRes = vRes + " </HotelReservations>"
            vRes = vRes + "</OTA_HotelResNotifRQ>"
```

En esta función básicamente estamos buscando los detalles de la reservación en la base de datos general, para que con cada detalle ir formando el documento XML. En el ejemplo anterior podemos ver ejemplos de las secciones principales de la función que genera estas reservaciones. Y a continuación tenemos la llamada al Web Service, donde pasa como parámetro el archivo generado.

```
    Private Sub ButtonSendOTA_80_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ButtonSendOTA_80.Click

        ' port 80

        Me.Cursor = Cursors.WaitCursor

        Dim vPath As String = "C:\Projects\.Net\Simulador\Simulador\XMLOTA\"
        Dim myService As New localhost_80.Service()
        Dim vEleRes As XmlElement = Nothing
        Dim vXMLRes As XmlDocument = New XmlDocument()
        Dim vRes As String = ""
        Dim vXMLReq As XmlDocument = New XmlDocument()

        vXMLReq.Load(vPath + "OTA_Request_New.xml")
        vEleRes = myService.processOTA(vXMLReq)

        vXMLReq.Load(vPath + "OTA_Request_Mod.xml")
        vEleRes = myService.processOTA(vXMLReq)

…

        vXMLRes.LoadXml(vEleRes.OuterXml())
        writeXMLLog("C:\Projects\.Net\Simulador\Simulador\XMLOTA\", "response", vXMLRes)

        Console.WriteLine(vXMLRes.InnerText())

        Me.Cursor = Cursors.Default

    End Sub
```

En la gráfica 3.1.1b podemos ver el ejemplo de la simulación del Portal que utiliza Thomson presentar sus reservaciones. Las funciones principales del Portal, es primero la obtención del usuario que esta asociado al hotel, para saber que reservaciones mostrar, luego tenemos una función que genera el html haciendo llamadas a funciones que obtienen los datos de cada reservación. Por ejemplo, aquí tenemos una función que obtiene los nombres de los pasajeros, y regresa la sección html que corresponde.

```
Public Function getNamesHTML(ByVal xHotel As String, _
                             ByVal xLocator As String) As String

    Dim vRes As String = ""
    Dim i As Integer = 0

    Dim vConnectionInt As New MySqlConnection()
    Dim vCommandInt As MySqlCommand = Nothing
    Dim vReaderInt As MySqlDataReader = Nothing

    vConnectionInt = openDB()
    vQuery = "select " _
             & "in_first_n, in_last_n, in_type " _
             & "from in_reserva, in_reserva_names " _
             & "where in_reserva.in_hotel = '" + xHotel + "' and " _
             & "in_reserva.in_hotel = in_reserva_names.in_hotel and " _
             & "in_reserva.in_locator = '" + xLocator + "' and " _
             & "in_reserva.in_reserva = in_reserva_names.in_reserva"
    vCommandInt = New MySqlCommand(vQuery, vConnectionInt)
    vReaderInt = vCommandInt.ExecuteReader()

    While vReaderInt.Read()
        vRes = vRes + "<tr align='center' class='displaytabletd2'>" & _
            "           <td></td>" & _
            "           <td>" + vReaderInt(0).ToString() + "</td>" & _
            "           <td>" + vReaderInt(1).ToString() + "</td>" & _
            "           <td>OWN-INS</td>" & _
            "           <td>" + vReaderInt(2).ToString() + "</td>" & _
            "         </tr>"
    End While

    vReaderInt.Close()
    closeDB(vConnectionInt)

    Return vRes
```



Gráfica 3.3.1b Pantalla principal del portal

3.3.2. Decodificadores

- **Web Service OTA XML**

A continuación se muestran las el método principal que espera reservaciones XML para la decodificación de este formato. Este método es el que decidirá que tipo de reservaciones provienen en el archivo XML, lo primero es guardar en un log el archivo proveniente, para después decidir que función llamara para su decodificación. Si existe algún error en el formato, se lanzara el sistema de notificación vía email.

```
    <WebMethod()> _
    Public Function processOTA(ByVal xmlMsg As XmlDocument) As XmlDocument

        Dim vXMLRes As XmlDocument = New XmlDocument()
        Dim vConnection As New MySqlConnection()
        Dim vQuery As String = ""
        Dim vRes As String = ""
        Dim vFunctions As New Functions()
        Dim vPath As String = "C:\Projects\.Net\OTA\logs\incoming\"
        Dim vNow As String = Now.ToString("yyyyMMdd_HHmmss")
        Dim vBookType As String = ""

        Try
            vFunctions.writeXMLlog(vPath, vNow, xmlMsg)

            vBookType = xmlMsg.DocumentElement.Attributes("ResStatus").InnerText()

            If vBookType.Equals("Commit") Then
                ' New
                vXMLRes = vFunctions.decodeNewModReserv(xmlMsg, vBookType)

            ElseIf vBookType.Equals("Modify") Then
                ' Modify
                vXMLRes = vFunctions.decodeNewModReserv(xmlMsg, vBookType)

            ElseIf vBookType.Equals("Cancel") Then
                ' Cancel
                vXMLRes = vFunctions.decodeCxlReserv(xmlMsg, vBookType)

            End If

            Return vXMLRes

        Catch ex As Exception
            vFunctions.SendEmail("Error OTA general en processOTA archivo " + vPath + vNow +
".xml", ex.Message())
            Return Nothing
        End Try

        Return vXMLRes

    End Function
```

La siguiente función esta compartida para reservaciones nuevas y modificadas, únicamente ya que la información es la misma, únicamente cambia el estatus. Esta función va buscando la información necesaria dentro del XML, para que una vez que tenga toda la información, entonces llamar una función que es la que inserta las reservaciones en la tabla de transferencias de la base de datos.

```
    Public Function decodeNewModReserv(ByVal xmlMsg As XmlDocument, _
                                       ByVal xAction As String) As XmlDocument



        Dim xmlResponse As XmlDocument = New XmlDocument()
        Dim stringResponse As String = ""
        Dim vNow As String = Now().ToString("yyyy-MM-ddTHH:mm:ss-05:00")
        Dim vAction As String = xAction.Replace("Commit", "Commited").Replace("Modify",
"Modified").Replace("Cancel", "Canceled")
…

        Try
…

            For i = 0 To nodeHotelReservation.Count() - 1
                ' Loop by reservations

                vStartTime = Now

                Try
                    vInpD = nodeHotelReservation(i).Attributes("CreateDateTime").InnerText()
                    vInpD = vInpD.Substring(0, vInpD.IndexOf("T"))
                Catch ex As Exception
                End Try
…
                resInsert = preInsertReserva(HotelCode, ResID_Value, CompanyName, xAction,
vInpD, vModD, Nothing, RoomTypeCode, RatePlanCode, _
                                             Comment, Adults, Childs, "0", NamePrefix,
GivenName, Surname, GuestAge, DateStart, DateEnd, AmountAfterTax, CurrencyCode, _
                                             ArrivalFlightDate, ArrivalFlightID,
DepartureFlightDate, DepartureFlightID, vStartTime)
…
        Catch ex As Exception

            xmlResponse.LoadXml("<error />")
            SendEmail("Error OTA en decodeNewModReserv hotel " + HotelCode + " locator " +
ResID_Value, ex.Message())
            Return xmlResponse

        End Try

End Function
```

- **Email**

A continuación se muestra un ejemplo de las función que regresa la fecha de salida de esta reservación, donde básicamente se esta buscando los índices de palabras para entonces sumarle la longitud extra de los elementos html.

```
public static String getDepartureDate(String xFile) {

	if (xFile.indexOf("CHECK-OUT DATE:") > 0) {

		int y, x = xFile.indexOf("CHECK-OUT DATE:") + 15;
		x = xFile.indexOf("<B>", x) + 3;
		y = xFile.indexOf("</B>", x);
		return xFile.substring(x,y).trim();

	} else {
		//Send NVL vRate does not exist
		return "NVL";
	}

}
```

Una función un poco más interesante, es la que nos regresa un arreglo de todos los pasajeros que vengan en la reservación. En este ejemplo, hacemos un ciclo del total de pasajeros, para este regresarlo en el arreglo haciendo búsquedas de los índices de las palabras claves.

```
public static String[] getPaxNames(String xFile, int xRooms, int xPax) {

		String[] vNames = new String[10];
		int x = 0,y = 0,vPeople = 0,vRoom = 1;

		if (xFile.indexOf("GUESTS:", 0) > 0) {

			for (int r = 1; r < xRooms + 1; r++) {

				for (int p = 0; p < xPax; p++) {

					if (r == 0 && p == 0) {
						x = xFile.indexOf("GUESTS:", 0) + 21;
					}

					x = xFile.indexOf("(ROOM "+r, x);
					int vTemp = x + 8;
					x = xFile.lastIndexOf("<B>",x) + 3;
					y = xFile.indexOf("</B>", x);

					vNames[vPeople] =
xFile.substring(x,y).trim().replace("'", "").replace("(", "").replace(")", "").replace(",",
"");
					vPeople++;
					x = vTemp;
				}

			}

		}

		return vNames;

	}
```

- **Web Scrapper**

La técnica utilizada para el Web Scrapper fue ir formando las direcciones URL, para después ir haciendo llamadas de estas. Primero se forma el URL principal para obtener la lista de todas las reservaciones en el portal, para después ir formando el URL de cada una de las reservaciones, hacer la llamada para ver el detalle de cada una, decodificar la información, y entonces insertarla en la tabla de transferencias de la base de datos. A continuación se muestra el método principal donde va formando los URL para después ir haciendo llamadas a estos.

```
Public Sub startProcess(ByVal xHotel As String)

    Dim vBaseURL As String = "http://localhost/portalUDLA/"
    Dim vRequest As System.Net.HttpWebRequest = Nothing
    Dim vResponse As System.Net.HttpWebResponse = Nothing
    Dim vResStreamHTML As System.IO.StreamReader = Nothing
    Dim vResHTML As String = Nothing
    Dim vStartTime As Date = Now
    Dim i As Integer = 0

    Try

        vListURL = vBaseURL + "ListUDLA.aspx?xHotel=" + xHotel + _
                              "&arrivalFromDate=27/09/2010" + _
                              "&arrivalToDate=26/12/2010"

        vDetailURL = vBaseURL + "DetailsUDLA.aspx?xHotel=" + xHotel + _
                                "&arrivalFromDate=27/09/2010" + _
                                "&arrivalToDate=26/12/2010" + _
                                "&xBooking="

        ' Send request and read the response
        vRequest = System.Net.HttpWebRequest.Create(vListURL)
        vResponse = vRequest.GetResponse()

        ' Read the response into the stream
        vResStreamHTML = New System.IO.StreamReader(vResponse.GetResponseStream)
        ' Get the response into a string
        vResHTML = vResStreamHTML.ReadToEnd()
        vResStreamHTML.Close()

        vResponse.Close()
        ' Gets the bookings from the HTML
        vBookings = getBoookings(vResHTML)

        Form1.ToolStripProgressBar1.Maximum = vBookings.Length() - 1

        For i = 0 To vBookings.Length() - 1
            ' Gets the detail for each booking
            vStartTime = Now
            vDetailURL = ""
            vDetailURL = vBaseURL + "DetailsUDLA.aspx?xHotel=" + xHotel + _
                                    "&arrivalFromDate=27/09/2010" + _
                                    "&arrivalToDate=26/12/2010" + _
                                    "&xBooking=" + vBookings(i).ToString()

            vRequest = System.Net.HttpWebRequest.Create(vDetailURL)
            vResponse = vRequest.GetResponse()
            vResStreamHTML = New System.IO.StreamReader(vResponse.GetResponseStream)
            vResHTML = vResStreamHTML.ReadToEnd()
            vResponse.Close()
            vResStreamHTML.Close()

            decodeUDLA(xHotel, vBookings(i), vResHTML, vStartTime)
            Form1.ToolStripProgressBar1.PerformStep()
        Next

    Catch ex As Exception
        Form1.Cursor = Cursors.Default
        Form1.ToolStripProgressBar1.Value = 0
        SendEmail("Error general en startProcess hotel " + xHotel, ex.Message())

    End Try

    Form1.ToolStripProgressBar1.PerformStep()
    Form1.ToolStripProgressBar1.Value = 0
    Form1.Cursor = Cursors.Default

End Sub
```

- **Flat File**

A continuación se muestra el método que busca la información en el archivo Flat File, basándose en las especificaciones se hacen los substrings de la cadena que contiene los detalles de la reservación.

```
    Public Sub processD(ByVal xline As String, _
                        ByVal xIATA As String, _
                        ByVal xhotel As String, _
                        ByVal xActDate As String, _
                        ByVal sr As StreamReader, _
                        ByVal xStartTime As Date)

        Dim longitud As String = xline.Length()

        Dim line As String = Nothing
        Dim arrayGuests() As String = New String(100) {}

        Try

            Dim type As String = xline.Substring(0, 1)                  ' Type
            Dim actionCode As String = xline.Substring(1, 1)            ' Action Code
            Dim closeOutIndicator As String = xline.Substring(2, 1)     ' Close-out
…
            Dim filler As String = xline.Substring(206, 28)             ' Filler
            Dim companyCode As String = xline.Substring(234, 3)         ' Company code
…
            If actionCode.Equals("B") Then
                inpD = xActDate
            ElseIf actionCode.Equals("M") Then
                modD = xActDate
            ElseIf actionCode.Equals("C") Then
                cxlD = xActDate
            End If
            arrayGuests = processG(xline, sr)

            If (rateDesc.Trim().Length() <= 0) Then
                rateDesc = "NVL"
            End If

            If (roomType.Trim().Length() <= 0) Then
                roomType = "NVL"
            End If

            preInsertReserva(xIATA, companyCode, xhotel.Trim(), xActDate, actionCode,
closeOutIndicator, replyCode, _
                             supplierDesc, serviceDesc, rateDesc.Trim(), specialService,
roomType, arrivalDate, _
                             numberOfNights, (TMTCReserva.Trim()), hotelConfirmation,
flightArrival, replyText, _
                             adultCount, juniorCount, childCount, filler, arrayGuests,
xStartTime)
        Catch ex As Exception
            SendEmail("Error FlatFile en processD hotel " + xhotel, ex.Message())
        End Try
    End Sub
```

3.4. Conclusiones

Para concluir, podemos decir que el sistema se integrará en una plataforma bajo Microsoft Windows XP, juntando los módulos desarrollados en Visual Basic.Net, y Java en conjunto de una base de datos común. Este sistema queda abierto para poder aumentar más módulos de Tour Operadores. El propósito de esta tesis es demostrar los cuatro método más utilizados por los ellos.