# MANUAL TÉCNICO

FARMA CRM

MANUAL TÉCNICO

SISTEMA DE GESTIÓN DE RELACIONES CON CLIENTES CRM.

El siguiente manual está diseñado para una administración y fácil mantenimiento del aplicativo.

El aplicativo está diseñado en Visual Studio 2012 con lenguaje de programación C#, con arquitectura de Entity Framework y como base de datos está SQL Server 2012.

Tiene un servidor de aplicaciones y un servidor de base de datos independientes, que se encuentran en las instalaciones de la empresa Farmaenlace Cía. Ltda., bajo la supervisión y mantenimiento del Área de Redes.

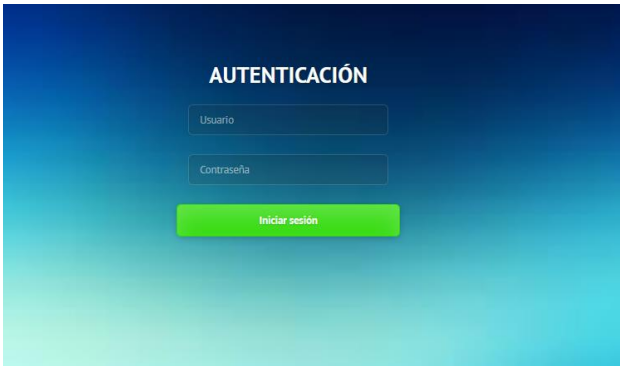
Esta es una aplicación web que se encuentra disponible en la Intranet de Farmaenlace.

Los módulos que contiene el aplicativo son:

- Comercial
- Notificaciones
- Reportes
- Análisis

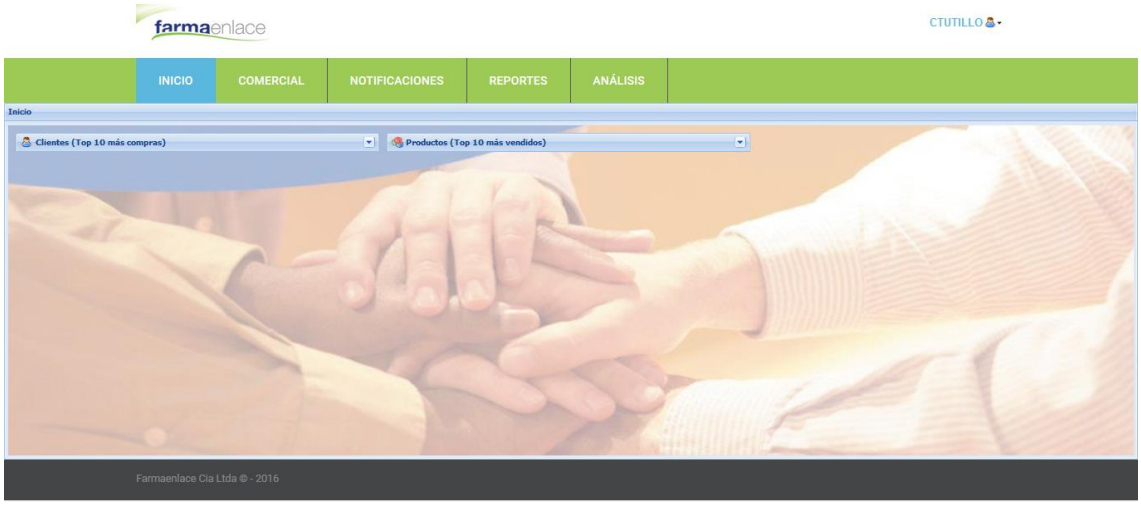
**Implementación del Aplicativo**

**1. Pantalla de Login** (login.aspx, login.cs).



```
/*****LOGIN DE LA APLICACION*****/


private void logueo()
{
    lblMensaje.Text = "";
    if (!String.IsNullOrEmpty(txtUserName.Text.Trim()) &&
    !String.IsNullOrEmpty(txtUserPwd.Text.Trim()))
    {
        string username = txtUserName.Text;
        string password = txtUserPwd.Text;
        string mensaje = String.Empty;
        try
        {
            wsLogin.WSEasyLoginSoapClient wsl = new wsLogin.WSEasyLoginSoapClient();
            String ACTDIR = wsl.GetParametroAccessPoint("EASYSEGURIDADNET", "ACTDIR");
           String AD_Path = ACTDIR.Split(',')[0];
           String AD_User = ACTDIR.Split(',')[1];
           String AD_Pass = ACTDIR.Split(',')[2];
           Boolean res = wsl.Authenticate(username, password, ref mensaje, AD_Path,
AD_User, AD_Pass);

           if (res)
           {

               wcfCRM.Service1Client objWcf=new wcfCRM.Service1Client();
               DataTable seguridad = objWcf.getPerfilesUsuario(username, "CRM");
               if (seguridad == null || seguridad.Rows.Count == 0)
                   lblMensaje.Text = "El usuario ingresado no tiene ningún privilegio.";
               else
               {
                   DataTable cedulaUsuario = objWcf.getCedulaUsuario(username);
                   string _cedula = cedulaUsuario.Rows[0]["cedula"].ToString();
                   if (_cedula.Length > 10)
                       _cedula = _cedula.Substring(0, 10);

                   else
                    {
                        Session["cedulaUsuario"] = _cedula;
                        Session["seguridad"] = seguridad;
                        Session["userLogin"] = username.ToLower();
                        Session["dtUser"] = cedulaUsuario;
                        X.Redirect("index.aspx");
                    }
               }
           }
           else
           {
               lblMensaje.Text = mensaje;
               txtUserPwd.Reset();
           }
       }
       catch (Exception ex)
       {
           this.msgAlertas("Excepción Login ..!!!","Autenticación fallida..!!<br/>"+
ex.Message+"<br/>Comuníquese con el departamento de sistemas.", MessageBox.Icon.ERROR, btnLogin);
        }
    }
    else
        lblMensaje.Text = "Todos los campos son obligatorios.";
}
```

## 2. Menú Principal




```
public void cargarFormulario(string opcion)
{
    if (opcion.Equals("camposD"))
    {
        contenido.AutoLoad.Url = "Clientes/frmCamposDinamicos.aspx";
        contenido.Title = "Comercial > Parametrización de Campos Dinámicos ";
    }
    else if (opcion.Equals("cliente"))
    {
        contenido.AutoLoad.Url = "Clientes/frmClientes.aspx";
        contenido.Title = "Comercial > Clientes";
    }

    else if (opcion.Equals("PlantillasMailAdm"))
    {
        contenido.AutoLoad.Url = "Mailing/frmPlantillas.aspx";
        contenido.Title = "Notificaciones > Administración de Plantillas";
    }
    else if (opcion.Equals("NotificacionesAdm"))
    {
        contenido.AutoLoad.Url = "Mailing/frmNotificacionesAdm.aspx";
        contenido.Title = "Notificaciones > Parametrización de Notificaciones";
    }

    else if (opcion.Equals("RptNotifGenerados"))
    {
        contenido.AutoLoad.Url = "Reportes/frmRptNotificaciones.aspx";
        contenido.Title = "Reportes > Notificaciones Generadas";
    }
    else if (opcion.Equals("OLAPClientes"))
    {
          contenido.AutoLoad.Url = "CuboClientes/frmCubo.aspx";
          contenido.Title = "Análisis > OLAP Clientes";
      }
      else if (opcion.Equals("OLAPMF"))
      {
          contenido.AutoLoad.Url = "CuboClientes/frmCuboMF.aspx";
          contenido.Title = "Análisis > OLAP Clientes MF ";
      }
```

```
      else
          contenido.Title = ".";

      this.contenido.AutoLoad.Mode = LoadMode.IFrame;
      this.contenido.AutoLoad.NoCache = true;
      this.contenido.LoadContent();

}
```

## MÉTODOS DEL MÓDULO COMERCIAL.

### SubMenú Clientes.

***Buscar Cliente*** (frmClientes.aspx.cs)

```
private void buscarCliente()
{
    try
    {
        wcfCRM.Service1Client objWcf1 = new wcfCRM.Service1Client();
        List<CRM_Clientes> lstClientes = new List<CRM_Clientes>();
        if (cbxBuscarCliente.Value.ToString().Equals("CI"))
        {
            objWcf1.Open();
            lstClientes = objWcf1.getClientesxCI(txtClienteBuscar.Text).ToList();
            objWcf1.Close();
        }
        else if (cbxBuscarCliente.Value.ToString().ToLower().Equals("name"))
        {
            objWcf1.Open();
            lstClientes = objWcf1.getClientes(txtClienteBuscar.Text).ToList();
            objWcf1.Close();
        }
        storeCliente.DataSource = lstClientes;
        storeCliente.DataBind();
        lstgridCLientes = lstClientes;
        if (lstClientes.Count <= 0)
            this.msgAlertas("Mensaje.", "No se encontraron coincidencias para el
parámetro buscado.", MessageBox.Icon.WARNING, btnBuscarCliente);
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR);
        storeCliente.DataSource = new DataTable();
        storeCliente.DataBind();
    }
}

// Métodos en la capa de servicio (CRM_ClientesServicio.cs)
public Entidades.CRM_Clientes getClientexCI(string cedula)
{
    clientesRepositorio.Inicializar();
    Entidades.CRM_Clientes objCliente = clientesRepositorio.Find(t => t.cli_cedula == cedula);
    return objCliente;
}
public List<Entidades.CRM_Clientes> getCliente(string nombres)
{
    clientesRepositorio.Inicializar();
    List<Entidades.CRM_Clientes> lstCliente = clientesRepositorio.Filter(t =>
t.cli_nombres.Contains(nombres)).ToList();
```

```
    return lstCliente;
}
```

***Actualizar Cliente***

```
try
{
    wcfCRM.Service1Client objWcf1 = new wcfCRM.Service1Client();

    DataTable dtcamposClientes = dtCamposfrm.Copy();// para los id de los ctrls
    CRM_InformacionClientes objInfoCli = new CRM_InformacionClientes();
    List<CRM_InformacionClientes> lstCli = new List<CRM_InformacionClientes>();
    List<CRM_Clientes> lstCliente = objWcf1.getClientesxCI(h_cedulaCli.Text).ToList();

    CRM_Clientes objCli = lstCliente[0];
    List<CRM_RelacionCamposFijos> lstRelacion = objWcf1.getRelacionCampos().ToList();
    string sqlUpdCli = "update CRM_Clientes set ";
    foreach (DataRow r in dtcamposClientes.Rows)
    {
        for (int i = 0; i < lstRelacion.Count; i++)
        {
            if (r["camd_codigo"].ToString() == lstRelacion[i].camd_codigo)
            {
                if (r["camd_codigo"].ToString() == "ctl1" || r["camd_codigo"].ToString()
== "ctl01" | lstRelacion[i].rcf_campo == "cli_cedula")
                    break;
                if (!String.IsNullOrEmpty(Page.Request[r["camd_codigo"].ToString()]))
                    sqlUpdCli += lstRelacion[i].rcf_campo + "='" +
Page.Request[r["camd_codigo"].ToString()] + "',";
                break;

            }
        }
    }
    sqlUpdCli = sqlUpdCli.Substring(0, sqlUpdCli.Length - 1);
    sqlUpdCli += " where cli_cedula='" + h_cedulaCli.Text + "'";

    foreach (DataRow r in dtcamposClientes.Rows)
    {
        if (Page.Request[r["camd_codigo"].ToString()] != null)
        {
            objInfoCli = new CRM_InformacionClientes();
            objInfoCli.cli_cedula = h_cedulaCli.Text;
            objInfoCli.camd_codigo = r["camd_codigo"].ToString();
            objInfoCli.infc_valor = Page.Request[r["camd_codigo"].ToString()];
            objInfoCli.infc_usuario_modifica = Session["userLogin"].ToString();
            lstCli.Add(objInfoCli);
        }
    }

    if (objWcf1.updateDataCliente(lstCli))
    {
                objWcf1.updateClienteCamposFijos(sqlUpdCli);
        this.buscarCliente();
        this.msgAlertas("Información.", "Registro actualizado con éxito.<br/>",
MessageBox.Icon.INFO, btnGuardarCambios1);
        windFrmCliente.Hide();
    }
    else
    {
```

```
        this.msgAlertas("Alerta...!!!", "Información del cliente no
actualizada.<br/><br/>Vuelva a intentar o comuníquese con el administrador del sistema.",
MessageBox.Icon.ERROR, btnGuardarCambios1);
    }

}
catch (Exception ex)
{
    this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR,
btnGuardarCambios1);
}
```

***Generar Plantilla***

```
protected void btnGenerarPlantilla_Click(object sender, EventArgs e)
{
    try
    {
        DataTable dtCamposClientes = dtCamposfrm.Copy();
        string json = "[{'ctl1':'CEDULA',";
        string valor, v1;

        foreach (DataRow r in dtCamposClientes.Rows)
        {
            valor = Request.Params["chk" + r["camd_codigo"].ToString()];
            if (valor != null)
            {
                for (int i = 0; i < lstCamposD.Count; i++)
                {
                    if (r["camd_codigo"].ToString().Equals(lstCamposD[i].camd_codigo))
                    {
                        json += "'" + r["camd_codigo"].ToString() + "':'" +
lstCamposD[i].camd_etiqueta + "',";
                        break;
                    }
                }
            }
            v1 = Request.Params[r["camd_codigo"].ToString()];
        }
        json = json.Substring(0, json.Length - 1);
        json += "}]";

        StoreSubmitDataEventArgs eSubmit = new StoreSubmitDataEventArgs(json, null);
        System.Xml.XmlNode xml = eSubmit.Xml;
        this.Response.Clear();
        //set the content type of the file to be downloaded (IF NEEDED)
        this.Response.ContentType = "application/vnd.ms-excel";//vnd.ms-excel
        this.Response.AppendHeader("Content-Disposition", "attachment;
filename=CRMPlantillaClientes.xls");
        this.Response.Charset = "utf-8";
        System.Xml.Xsl.XslCompiledTransform xtExcel = new
System.Xml.Xsl.XslCompiledTransform();
        xtExcel.Load(Server.MapPath("Excel.xsl"));
        xtExcel.Transform(xml, null, this.Response.OutputStream);
        Response.Flush();
        this.Response.End();
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", "Vuelva a intentar o comuniquese con el
administrador del sistema <br/>" + ex.Message, MessageBox.Icon.ERROR);
    }
```

```
}
```

***Subir Archivo Clientes*** (frmImportClientes.aspx.cs)

```
protected void btnUpArchivo_Click(object sender, DirectEventArgs e)
{
    OleDbConnection objCon=new OleDbConnection();

    try
    {
        if (fileUp.HasFile)
        {
            string extension = System.IO.Path.GetExtension(fileUp.FileName);
            if (extension == ".xls" || extension == ".xlsx")
            {
                h_saveFile.Value = "file";
                DataTable dtArchivo = new DataTable();

                HttpBrowserCapabilities bc = Request.Browser;
                string navegador = bc.Browser;
                string fileName = fileUp.FileName;

                string path = Server.MapPath("~/doc/");

                HttpFileCollection files = Request.Files;
                for (int i = 0; i <= files.Count - 1; i++)
                {
                    if (files.AllKeys[i] == "fileUp-file")
                    {
                        HttpPostedFile postedFile = files[i];
                        postedFile.SaveAs(Server.MapPath("~/doc/") +
System.IO.Path.GetFileName(postedFile.FileName));
                        path = Server.MapPath("~/doc/") +
System.IO.Path.GetFileName(postedFile.FileName);
                        break;
                    }
                }
                string cadena;
                cadena = @"Provider=Microsoft.ACE.OLEDB.12.0;" +
                                    @"Data Source=" + path + ";" + @"Extended Properties="
+ '"' + "Excel 12.0;HDR=YES" + '"';
                objCon = new OleDbConnection(cadena);
                objCon.Open();
                DataTable dtHojas = objCon.GetOleDbSchemaTable(OleDbSchemaGuid.Tables, null);
                string namehoja1 = dtHojas.Rows[0]["TABLE_NAME"].ToString();

                string strSQL = "SELECT * FROM [" + namehoja1 + "]";
                OleDbDataAdapter da = new OleDbDataAdapter(strSQL, objCon);
                da.Fill(dtArchivo);
                objCon.Close();

                DataTable dtaux = dtArchivo.Copy();
                dtArchivo.Columns.Add("ERROR");
                int f = 0;
                foreach (DataColumn c in dtArchivo.Columns)
                {
                    storeDataImport.AddField(new RecordField(c.ColumnName));
                    storeErrores.AddField(new RecordField(c.ColumnName));
                    storeDataImport.ClearMeta(); storeErrores.ClearMeta();
                    string name = dtArchivo.Rows[0][f].ToString() ;
                    gridDataImport.ColumnModel.Columns.Add(new Column { ColumnID =
c.ColumnName, DataIndex = c.ColumnName, Header = name, Menu
```

```
                    gridErrores.ColumnModel.Columns.Add(new Column { ColumnID =
c.ColumnName, DataIndex = c.ColumnName, Header = name, MenuDis
                    f++;
                }
                object []row = dtArchivo.Rows[0].ItemArray;
                dtArchivo.Rows.RemoveAt(0);
                this.validarExcel(dtArchivo,row);
                storeDataImport.DataSource = dtArchivo;
                storeDataImport.DataBind();
                gridDataImport.Reconfigure();
                gridErrores.Reconfigure();
            }
            else
                this.msgAlertas("Advertencia...!!!", "Seleccione un archivo excel y
vuelva a intentar.", MessageBox.Icon.ERROR, btnUpArchivo);
        }
        else
            this.msgAlertas("Advertencia...!!!", "No ha seleccionado ningun archivo.",
MessageBox.Icon.ERROR, btnUpArchivo);
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR);
        objCon.Close();
    }
}
```

**Submenú Campos Dinámicos**

***Crear Nuevo Campo*** (frmCamposDinamicos.aspx.cs)

```
protected void btnSaveNewCampoD_Click(object sender, DirectEventArgs e)
{
    try
    {
        if (cbxTipoCampo.Value != null)
        {
            if (txtEtiqueta.Text.Trim() != "")
            {
                CRM_CamposDinamicos camposD = new CRM_CamposDinamicos();
                camposD.camd_codigo = txtIDCampo.Text;
                camposD.camd_etiqueta = txtEtiqueta.Text.Trim().ToUpper();
                camposD.tipd_codigo = cbxTipoCampo.Value.ToString();
                camposD.camd_orden = "0";
                camposD.camd_estado = "A";
                camposD.camd_usuario_creacion = Session["userLogin"].ToString();
                camposD.camd_tipo = "D";
                camposD.camd_editable = "N";
                camposD.camd_visible = "N";
                if (chkEditable.Checked)
                    camposD.camd_editable = "S";
                if (chkVisible.Checked)
                    camposD.camd_visible = "S";

                objWCF = new Service1Client();
                bool flagRes = false;
                if (cbxTipoCampo.Value.ToString() == "combo")
                {
```

```
                    string grid = e.ExtraParams["Values"];
                    Dictionary<string, string>[] filas =
JSON.Deserialize<Dictionary<string, string>[]>(grid);
                    if (filas.Length >= 1)
                    {
                        CRM_ParametrosCampos opcionesCBX = new CRM_ParametrosCampos();
                        List<CRM_ParametrosCampos> lstOpcionesCBX = new
List<CRM_ParametrosCampos>();
                        foreach (Dictionary<string, string> row in filas)
                        {
                            opcionesCBX = new CRM_ParametrosCampos();
                            opcionesCBX.parmc_codigo = "param" +
row["ordencbx"].ToString();
                            opcionesCBX.camd_codigo = txtIDCampo.Text;
                            opcionesCBX.parmc_valor = row["opcionescbx"].ToString();
                            opcionesCBX.parmc_orden =
Convert.ToInt32(row["ordencbx"]).ToString();
                            opcionesCBX.parmc_estado = "A";
                            lstOpcionesCBX.Add(opcionesCBX);
                        }

                        flagRes = objWCF.grabarCampoDinamico(camposD, lstOpcionesCBX);

                    }
                    else
                    {
                        this.msgAlertas("Alerta...!!!", "El tipo de dato seleccionado
requiere al menos un valor para procesar la información.<br/>Campo no almacenado. ",
MessageBox.Icon.ERROR, btnSaveNewCampoD);
                        return;
                    }
                }
                else if (cbxTipoCampo.Value.ToString() == "text")
                {

                    if (txtTextoDefault.Text.Trim() != "")
                    {
                        List<CRM_ParametrosCampos> lstParam = new List<CRM_ParametrosCampos>();
                        CRM_ParametrosCampos paramDefault = new CRM_ParametrosCampos();
                        paramDefault.parmc_codigo = "param";
                        paramDefault.camd_codigo = txtIDCampo.Text;
                        paramDefault.parmc_valor = txtTextoDefault.Text.Trim();
                        paramDefault.parmc_orden = "0";
                        paramDefault.parmc_estado = "A";
                        lstParam.Add(paramDefault);
                        flagRes = objWCF.grabarCampoDinamico(camposD, lstParam);
                    }
                    else
                    {
                        flagRes = objWCF.grabarCampoDinamico(camposD, null);
                    }

                }
                else if (cbxTipoCampo.Value.ToString() == "num")
                {
                    if (txtTextoDefault.Text.Trim() != "")
                    {
                        List<CRM_ParametrosCampos> lstParam = new List<CRM_ParametrosCampos>();
                        CRM_ParametrosCampos paramDefault = new CRM_ParametrosCampos();
                        paramDefault.parmc_codigo = "param";
                        paramDefault.camd_codigo = txtIDCampo.Text;
                        paramDefault.parmc_valor = txtNumDefault.Text.Trim();
                        paramDefault.parmc_orden = "0";
```

```
                        paramDefault.parmc_estado = "A";
                        lstParam.Add(paramDefault);
                        flagRes = objWCF.grabarCampoDinamico(camposD, lstParam);
                    }
                    else
                    {
                        flagRes = objWCF.grabarCampoDinamico(camposD, null);
                    }

                }
                else if (cbxTipoCampo.Value.ToString() == "date")
                {
                    string fecha = "";
                    if (dateDefault.Text == "0001-01-01 0:00:00" || dateDefault.Text ==
"1900-01-01 0:00:00")
                        fecha = "1900-01-01";
                    else
                        fecha = dateDefault.Text.Trim(); ;

                    List<CRM_ParametrosCampos> lstParam = new List<CRM_ParametrosCampos>();
                    CRM_ParametrosCampos paramDefault = new CRM_ParametrosCampos();
                    paramDefault.parmc_codigo = "param";
                    paramDefault.camd_codigo = txtIDCampo.Text;
                    paramDefault.parmc_valor = fecha;
                    paramDefault.parmc_orden = "0";
                    paramDefault.parmc_estado = "A";
                    lstParam.Add(paramDefault);
                    flagRes = objWCF.grabarCampoDinamico(camposD, lstParam);

                }
                if (flagRes)
                {
                    this.msgAlertas("Información.", "Registro guardado con éxito.",
MessageBox.Icon.INFO, btnSaveNewCampoD);
                    this.cbxTipoCampo_Select();
                    this.showListCamposD();
                    windCrearCampos.Hide();
                }
                else
                    this.msgAlertas("Error...!!!", "Registro no
guardado.<br/><br/>Compruebe que el campo (etiqueta) que esta ingresando no
exista.<br/>Vuelva a intentar; si continua el problema comuníquese con el administrador
del sistema.", MessageBox.Icon.ERROR, btnSaveNewCampoD);
            }
            else
                this.msgAlertas("Alerta...!!!", "Ingrese un nombre de etiqueta para el
nuevo campo.", MessageBox.Icon.ERROR, btnSaveNewCampoD);
        }
        else
            this.msgAlertas("Alerta...!!!", "Seleccione el tipo de dato para el nuevo
campo.", MessageBox.Icon.ERROR, btnSaveNewCampoD);
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR, btnSaveNewCampoD);
    }
}
// Métodos en la Capa de servicio(CRM_CamposDinamicosServicio.cs)
public bool grabarNewCampos(Entidades.CRM_CamposDinamicos newCampoD)
{
    try
    {
        CRM_CamposDinamicos camposD = new CRM_CamposDinamicos();
```

```
        int numTotalCampos = campoDinamicoRepositorio.Count;
        //long i = campoDinamicoRepositorio.Max(t => Convert.ToInt64(t.camd_orden));
        camposD.camd_codigo = newCampoD.camd_codigo;
        camposD.camd_etiqueta = newCampoD.camd_etiqueta;
        camposD.tipd_codigo = newCampoD.tipd_codigo;
        camposD.camd_orden = numTotalCampos.ToString();
        camposD.camd_estado = newCampoD.camd_estado;
        camposD.camd_usuario_creacion = newCampoD.camd_usuario_creacion;
        camposD.camd_tipo = newCampoD.camd_tipo;
        camposD.camd_editable = newCampoD.camd_editable;
        camposD.camd_visible = newCampoD.camd_visible;

        int filasAfect = campoDinamicoRepositorio.Create(camposD);
        if (filasAfect > 0)
            return true;
        return false;
    }
    catch (Exception ex)
    {
        log.GrabarLogError("CAMPOSDIN_SERVICIO.  grabarNewCampos(obj) // "+ex.Message);
        return false;
    }
}

// Actualizar
public bool updateCampoDinamico(CRM_CamposDinamicos campoDinamico)
{
    try
    {
        CRM_CamposDinamicos camposD = campoDinamicoRepositorio.Find(t => t.camd_codigo ==
campoDinamico.camd_codigo);
        if (camposD != null)
        {
            camposD.camd_etiqueta = campoDinamico.camd_etiqueta;
            if (campoDinamico.camd_estado != null && campoDinamico.camd_estado != "")
                camposD.camd_estado = campoDinamico.camd_estado;
            camposD.camd_editable = campoDinamico.camd_editable;
            camposD.camd_visible = campoDinamico.camd_visible;
            if (campoDinamicoRepositorio.Update(camposD) > 0)
                return true;
        }
    }
    catch (Exception ex)
    {
        log.GrabarLogError("updateCampoDinamico(obj) //  "+ex.Message+ex.InnerException);
        return false;
    }
    return false;
}
```

## MÉTODOS DEL MÓDULO NOTIFICACIONES

**Submenú Plantillas** (frmPlantillas.aspx.cs)

```
private void showListPlantillas()
{
    try
    {
        objCRM = new wcfCRM.Service1Client();
        List<CRM_PlantillasMail> lstPlantillas = new List<CRM_PlantillasMail>();
```

```
        objCRM.Open();
        lstPlantillas = objCRM.getAllPlantillas().ToList();
        objCRM.Close();
        lstPlantillas = lstPlantillas.OrderBy(t => t.plm_nombre).OrderBy(t =>
t.plm_estado).ToList();
        storeListaPlantilla.DataSource = lstPlantillas;
        //storeListaPlantilla.Sort("plm_nombre", Ext.Net.SortDirection.ASC);
        storeListaPlantilla.DataBind();
        pgListPlantilla.SetPageIndex(1);
        pgListPlantilla.DataBind();
        lstPlantillas_gen = lstPlantillas;
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR);
        objCRM.Close();
    }
}
//crear vista previa de la plantilla
protected void crearVistaHTML(string body,string fileName)
{
    string path = Server.MapPath("~/Mailing/Plantillas/temp/");
    foreach (string fichero in Directory.GetFiles(path))  //, "*.html"
        File.Delete(fichero);

    StreamWriter strWriter = File.CreateText(path + fileName);
    strWriter.WriteLine("<html>");
    strWriter.WriteLine("<head>");
    strWriter.WriteLine("<meta http-equiv='Content-Type' content='text/html; charset=utf-8'>");
    strWriter.WriteLine("<title>" + txtNombrePlantilla.Text + "</title>");
    //strWriter.WriteLine("");
    strWriter.WriteLine("</head>");
    strWriter.WriteLine("<body>");
    strWriter.WriteLine(body);
    strWriter.WriteLine("</body>");
    strWriter.WriteLine("</html>");
    strWriter.Close();

    X.Msg.Show(new MessageBoxConfig
    {
        Message = "Generando plantilla. Espere...",
        ProgressText = "Saving...",
        Width = 300,
        Wait = true,
        WaitConfig = new WaitConfig { Interval = 150 },
        IconCls = "ext-mb-download",
        AnimEl = this.mnVPDatos.ClientID
    });

    this.ResourceManager1.AddScript("setTimeout(function () { Ext.MessageBox.hide(); " +

"window.open('../Mailing/Plantillas/temp/" + fileName + "','_blank', 'status = yes, too


}
```

**Guardar Plantilla**

```
protected void btnGuardarPlantilla_Click(object sender, DirectEventArgs e)
{
    objCRM = new Service1Client();
```

```
    string msgWCF="";
    try
    {
        msgAlerta = "";
        CRM_PlantillasMail objPlantillas = this.leerPlantilla(0);
        if (objPlantillas != null)
        {
            objCRM.Open();
            msgWCF = objCRM.grabarPlantilla(objPlantillas);
            objCRM.Close();
            if (msgWCF.Equals("ok"))
            {
                this.msgAlertas("Información...", "Registro guardado con éxito.",
MessageBox.Icon.INFO, btnGuardarPlantilla);
                this.limpiarControlesPlantilla();
                this.showListPlantillas();
                windFrmPlantilla.Hide();
            }
            else
                this.msgAlertas("Error...!!!", "<b>Registro no
guardado.</b><br/><br/>Comprobar que el nombre de la plantilla no exista.<br/Vuelva a
intentar; si continua el problema comuníquese con el administrador del
sistema.<br/><br/>" + msgWCF, MessageBox.Icon.ERROR, btnGuardarPlantilla);
        }
        else
        {
            if (msgAlerta != "")
                this.msgAlertas("Alerta...!!!", msgAlerta, MessageBox.Icon.WARNING,
btnGuardarPlantilla);
            else
                this.msgAlertas("Mensaje...!!!", "No se ha podido obtener la información
del formulario.<br/>Recargue la página y vuelva a intentar.", MessageBox.Icon.WARNING);
        }
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", msgWCF + " " + ex.Message,
MessageBox.Icon.ERROR, btnGuardarPlantilla);
    }

// Método grabar en capa de servicio (CRM_PlantillasMailServicio.cs)
public List<CRM_PlantillasMail> getAllPlantillas()
{
    plantillasMRepositorio.Inicializar();
    List<CRM_PlantillasMail> lstPlantillas = plantillasMRepositorio.All().ToList();
    return lstPlantillas;
}

public CRM_PlantillasMail getPlantilla(int codPlantilla)
{
    plantillasMRepositorio.Inicializar();
    CRM_PlantillasMail objPlantilla = plantillasMRepositorio.Find(t => t.plm_codigo ==
codPlantilla);
    return objPlantilla;
    throw new NotImplementedException();
}
public string grabarPlantilla(CRM_PlantillasMail objPlantilla)
{
    try
    {
        CRM_PlantillasMail objNewPlantilla = new CRM_PlantillasMail();
        objNewPlantilla.plm_nombre = objPlantilla.plm_nombre;
```

```
        objNewPlantilla.plm_descripcion = objPlantilla.plm_descripcion;
        objNewPlantilla.plm_asunto = objPlantilla.plm_asunto;
        objNewPlantilla.plm_filehtml = objPlantilla.plm_filehtml;
        objNewPlantilla.plm_estado = objPlantilla.plm_estado;
        objNewPlantilla.plm_tipo = objPlantilla.plm_tipo;
        objNewPlantilla.plm_fecha_creacion = DateTime.Now;
        objNewPlantilla.plm_usuario_creacion = objPlantilla.plm_usuario_creacion;
        objNewPlantilla.plm_modificacion = objPlantilla.plm_modificacion;
        int filasAfect = plantillasMRepositorio.Create(objNewPlantilla);
        if (filasAfect > 0)
            return "ok";
        return "false";
    }
    catch (Exception ex)
    {
        log.GrabarLogError("Metod: grabarPlantilla(obj) // " + ex.Message+"\n *
"+ex.InnerException.InnerException.Message);
        return ex.Message;
    }
    throw new NotImplementedException();
}
```

## Submenú Notificaciones

### Grabar Notificación

```
protected void btnGuardarNotif_Click(object sender, DirectEventArgs e)
{
    try
    {
        addMSG = "";
        objWcf = new wcfCRM.Service1Client();
        CRM_Notificaciones objNotif = this.leerFRMNotif(e,true);
        if (objNotif != null && !String.IsNullOrEmpty(objNotif.not_asunto))
        {
            objWcf.Open();
            string msgWCF = objWcf.grabarNotificaciones(objNotif);
            objWcf.Close();
            if (msgWCF.Equals("ok"))
            {
                this.showListaNotificaciones();
                windFrmNotificaciones.Hide();
                this.limpiarFRMNotif();
                this.msgAlertas("Información.", "Registro guardado con éxito." + addMSG,
MessageBox.Icon.INFO);
            }
            else
                this.msgAlertas("Error...!!!", "<b>Registro no guardado.</b><br/>Vuelva a
intentar; si continua el problema comuníquese con el administrador del
sistema.<br/><br/>" + msgWCF, MessageBox.Icon.ERROR, btnGuardarNotif);
        }
    }
    catch (Exception ex)
    { this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR); }
}

private CRM_Notificaciones guardarNotifProgramadas(string grdDestinatarios)
{

    DateTime dateFechaFin = DateTime.Parse("1900-01-01");
    CRM_Notificaciones objNotif = new CRM_Notificaciones();
    if (chkgrpDias.CheckedItems.Count > 0)
```

```
    {
        if ((dateFInicio.Text.Trim() != "") && (dateFInicio.Text != "01-01-0001 0:00:00")
&& (dateFInicio.Text != "0001-01-01 0:00:00"
        {
            if ((dateFFin.Text != "0001-01-01 0:00:00") && (dateFFin.Text != "01-01-0001 0:00:00"))
            {
                dateFechaFin = dateFFin.SelectedDate;
                if (dateFInicio.SelectedDate > dateFFin.SelectedDate)
                {
                    this.msgAlertas("Alerta...!!!", "La Fecha_Inicio no puede ser mayor
que la Fecha_Fin.  ", MessageBox.Icon.ERROR, btnGuardarNotif);
                    return null;
                }
            }
            if (!String.IsNullOrEmpty(txtPlantilla.Text.Trim()) &&
!String.IsNullOrEmpty(h_IDPlantilla.Text.Trim()))
            {

                CRM_NotificacionesFrecuencia[] arrayFrec = new
CRM_NotificacionesFrecuencia[chkgrpDias.CheckedItems.Count];
                objNotif.not_codigo = 0;
                objNotif.not_asunto = txtAsunto.Text;
                objNotif.plm_codigo = Convert.ToInt32(h_IDPlantilla.Text);
                objNotif.not_hora = timeNotifProg.Text;//TimeSpan.Parse()
                objNotif.not_fecha_creacion = DateTime.Now;
                objNotif.not_fecha_inicio = dateFInicio.SelectedDate;
                objNotif.not_fecha_fin = dateFechaFin;//dateFFin.SelectedDate;
                objNotif.not_estado = "A";
                objNotif.not_tipo = "PROGRAMADO";
                objNotif.not_fecha_creacion = DateTime.Now;
                objNotif.not_usuario_creacion = Session["userLogin"].ToString();
                objNotif.not_modificacion = DateTime.Now.ToString() + "|" +
Session["userLogin"].ToString();


                List<Checkbox> lst = chkgrpDias.CheckedItems;
                for (int i = 0; i < chkgrpDias.CheckedItems.Count; i++)
                {
                    arrayFrec[i] = new CRM_NotificacionesFrecuencia();
                    arrayFrec[i].notfr_dia_semana = lst[i].Name.ToString().ToUpper();
                }
                objNotif.CRM_NotificacionesFrecuencia = arrayFrec.ToList();

                if (grdDestinatarios.Length > 0)
                {
                    Dictionary<string, string>[] filas =
JSON.Deserialize<Dictionary<string, string>[]>(grdDestinatarios);
                    CRM_NotificacionesClientes[] arrayDestinatarios = new
CRM_NotificacionesClientes[filas.Count()];
                    int i = 0;
                    int contSinMail=0;
                    foreach (Dictionary<string, string> row in filas)
                    {
                        if (row["correo"].ToString() != "" && row["correo"].ToString() != "N/A")
                        {
                            arrayDestinatarios[i] = new CRM_NotificacionesClientes();
                            arrayDestinatarios[i].cli_cedula = row["cedula"].ToString();
                            arrayDestinatarios[i].notcl_estado = "C";
                            i++;
                        }
                        else
                            contSinMail = contSinMail + 1;
                    }
```

```
                    CRM_NotificacionesClientes[] arrayDest = new
CRM_NotificacionesClientes[i];
                    for (int k = 0; k < i; k++)
                    {
                        arrayDest[k] = new CRM_NotificacionesClientes();
                        arrayDest[k].cli_cedula = arrayDestinatarios[k].cli_cedula;
                        arrayDest[k].notcl_estado = arrayDestinatarios[k].notcl_estado;
                    }
                    objNotif.CRM_NotificacionesClientes = arrayDest.ToList();
                    if (contSinMail > 0)
                        addMSG = "<br/><b>NOTA:</b><br>Los registros sin correo no fueron
almacenados: " + contSinMail;
                }
            }
            else
                this.msgAlertas("Alerta...!!!", "No ha seleccionado ninguna plantilla.",
MessageBox.Icon.ERROR);
        }
        else
            this.msgAlertas("Alerta...!!!", "Seleccione una fecha de inicio válida.",
MessageBox.Icon.ERROR);
    }
    else
        this.msgAlertas("Alerta...!!!", "No ha seleccionado ningún día de la semana.",
MessageBox.Icon.ERROR);
    return objNotif;
}
```

**Ejecutar Notificación**

```
public void esSiEjectNot
{
    try
    {
        objWcf = new Service1Client();

        if (crearFile(objNotificacion.CRM_PlantillasMail))
        {
                        string strHtml =
this.leerFile(objNotificacion.CRM_PlantillasMail.plm_codigo.ToString());
            DataTable dtDest = new DataTable();
            List<CRM_CamposDinamicos> lstCamposD = objWcf.getAllCamposD().ToList();
            if (objNotificacion.not_tipo.ToUpper() == "PROGRAMADO")
            {
                dtDest =
objWcf.getDestinatariosNotifProgramados(objNotificacion.not_codigo);
            }
            else if (objNotificacion.not_tipo.ToUpper() == "AUTOMATICO")
            {
                List<CRM_NotificacionesCamposFecha> objCamposFec =
objNotificacion.CRM_NotificacionesCamposFecha.ToList();

                dtDest =
objWcf.getDestinatariosNotifAutom(objCamposFec[0].notcf_agregar_dias,
objCamposFec[0].camd_codigo);
            }
            string strAsuntoMail = objNotificacion.CRM_PlantillasMail.plm_asunto;
            string msgErrorDestin = "";// para almacenar los destinatarios q no se envio
el mail
            msgErrorDestin += "<tr>";
            msgErrorDestin += "<td> <b> CEDULA </b> </td>";
            msgErrorDestin += "<td> <b> NOMBRES </b> </td>";
```

```
            msgErrorDestin += "<td> <b> OBSERVACIÓN </b> </td>";
            msgErrorDestin += "</tr>";
            DataTable dtErroresDest = new DataTable();
            dtErroresDest.Columns.Add("CEDULA");
            dtErroresDest.Columns.Add("NOMBRES");
            dtErroresDest.Columns.Add("OBSERVACION");
            if (dtDest.Rows.Count > 0)
            {
                foreach (DataRow r in dtDest.Rows)
                {
                    string etiqueta = "";
                    string bodyHtml = strHtml;
                    string correoDestinat = "";
                    correoDestinat = r["correo"].ToString();
                    if (correoDestinat != "")
                    {
                        try
                        {
                            for (int i = 0; i < lstCamposD.Count; i++)
                            {
                                etiqueta = lstCamposD[i].camd_etiqueta;
                                if (bodyHtml.Contains("$" + etiqueta + "$"))
                                {
                                    bodyHtml = bodyHtml.Replace(("$" + etiqueta + "$"),
r[etiqueta].ToString());
                                }
                            }
                        }
                        catch (Exception ex)
                        {

                            dtErroresDest.Rows.Add(new object[] { r["cedula"].ToString(),
r["nombres"].ToString(), "Campo:" + etiqueta +" "+ ex.Message });
                            msgErrorDestin += "<tr>";
                            msgErrorDestin += "<td>" + r["cedula"].ToString() + " </td>"
+ "<td>" + r["nombres"].ToString() + " </td>" + "<td>" + "Campo:" + etiqueta+ " " +
ex.Message + " </td>";
                            msgErrorDestin += "</tr>";
                        }

                        string cc = "";
                        string adjunto = "";
                        string resultMail = this.enviarMail(correoDestinat, cc,
strAsuntoMail, bodyHtml, adjunto);

                        if (!resultMail.ToLower().Equals("ok"))
                        {
                            dtErroresDest.Rows.Add(new object[] { r["cedula"].ToString(),
r["nombres"].ToString(), correoDestinat + " - " + resultMail });
                            msgErrorDestin += "<tr>";
                            msgErrorDestin += "<td>" + r["cedula"].ToString() + " </td>"
+ "<td>" + r["nombres"].ToString() + " </td>" + "<td>" + correoDestinat + " - " +
resultMail + " </td>";
                            msgErrorDestin += "</tr>";
                        }

                    }
                    else
                    {
                        dtErroresDest.Rows.Add(new object[] { r["cedula"].ToString(),
r["nombres"].ToString(), " Sin dirección de correo" });
                        msgErrorDestin += "<tr>";
```

```
                        msgErrorDestin += "<td>" + r["cedula"].ToString() + " </td>" +
"<td>" + r["nombres"].ToString() + " </td>" + "<td> Sin dirección de correo </td>";
                        msgErrorDestin += "</tr>";
                    }
                }

                    //enviar correo al userLogin
                    //enviar correo al adminCRM que se ha enviado la notificación
                if (dtErroresDest.Rows.Count > 0)
                {
                    storeErrores.DataSource = dtErroresDest;
                    storeErrores.DataBind();
                    windErroresMail.Title += " - " + h_notifAsunto.Text;
                    windErroresMail.Show();
                }
                this.msgAlertas("Información...", "Proceso terminado correctamente.",
MessageBox.Icon.INFO);
                //Mail información de envio de notificaciones por el usuario
                DataTable dtUser = (DataTable)Session["dtUser"];
                string strHtml_resp = Properties.Resources.mailErrorNotif;
                strHtml_resp = strHtml_resp.Replace("@notificacion", h_notifAsunto.Text);
                strHtml_resp = strHtml_resp.Replace("@usuario",
dtUser.Rows[0]["nombres"].ToString());
                strHtml_resp = strHtml_resp.Replace("@filasClientes", msgErrorDestin);
                this.enviarMail(dtUser.Rows[0]["email"].ToString(),
Properties.Settings.Default.correoAdm, "[CRM]-Ejecución de Notificaciones Manuales.",
strHtml_resp, "");
            }
            else
                this.msgAlertas("Alerta...!!!", "La notificación no contiene
destinatarios. ", MessageBox.Icon.WARNING);

        }
        else
            this.msgAlertas("Alerta...!!!", "Plantilla no generada.<br/>, Favor volver a
cargar el formulario y volver a intentar.<br/>Si el problema continua comuníquese con el
administrador del sistema. ", MessageBox.Icon.ERROR);
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR);
    }

}

// Métodos capa de servicio (CRM_NotificacionesServicio.cs)
public List<CRM_Notificaciones> getAllNotificaciones()
{
    notifRepositorio.Inicializar();
    List<CRM_Notificaciones> lstNotif = notifRepositorio.All().ToList();
    return lstNotif;
}
public CRM_Notificaciones getNotificacion(int codigoNotf)
{
    notifRepositorio.Inicializar();
    CRM_Notificaciones objNotif = notifRepositorio.Find(t => t.not_codigo == codigoNotf);
    return objNotif;
}

public string grabarNotificaciones(CRM_Notificaciones objNotificaciones)
{
    try {
```

```
        CRM_Notificaciones newNotificacion = new CRM_Notificaciones();
        newNotificacion.not_codigo = 0;
        newNotificacion.plm_codigo = objNotificaciones.plm_codigo;
        newNotificacion.not_asunto = objNotificaciones.not_asunto;
        newNotificacion.not_hora = objNotificaciones.not_hora;
        newNotificacion.not_fecha_inicio = objNotificaciones.not_fecha_inicio;
        newNotificacion.not_fecha_fin = objNotificaciones.not_fecha_fin;
        newNotificacion.not_estado = objNotificaciones.not_estado;
        newNotificacion.not_tipo = objNotificaciones.not_tipo;
        newNotificacion.not_fecha_creacion = DateTime.Now;
        newNotificacion.not_usuario_creacion = objNotificaciones.not_usuario_creacion;
        newNotificacion.not_modificacion = objNotificaciones.not_modificacion;
        newNotificacion.CRM_NotificacionesFrecuencia = objNotificaciones.CRM_NotificacionesFrecuencia;
        newNotificacion.CRM_NotificacionesClientes = objNotificaciones.CRM_NotificacionesClientes;
        newNotificacion.CRM_NotificacionesCamposFecha = objNotificaciones.CRM_NotificacionesCamposFecha;
        int filasAfect = notifRepositorio.Create(newNotificacion);
        if (filasAfect > 0)
            return "ok";
        return "false";
    }
    catch (Exception ex)
    {
        log.GrabarLogError("Metd:grabarNotfc(obj) // \n" + ex.Message +
ex.InnerException.InnerException.Message);
        return ex.InnerException.Message;
    }
}
```

## MÉTODOS MÓDULO REPORTES

**Generar Reporte Notificaciones** (frmRptNotificaciones.aspx.cs)

```
protected void btnGenerarReport_Click(object sender, DirectEventArgs e)
{
    try
    {
        SelectedListItemCollection itemNotif = cbxNotificacion.SelectedItems;
        string codigoNotif = "";
        foreach (SelectedListItem s in itemNotif)
        {
            if (s.Value.ToLower().Equals("todos"))
            {
                codigoNotif = "";
                break;
            }
            //string txt = s.Text;//string val = s.Value;
            codigoNotif += s.Value + ",";
        }
        if (codigoNotif.Length > 0)
            codigoNotif = codigoNotif.Substring(0, (codigoNotif.Length-1));

        if (itemNotif.ToList().Count > 0)
        {
            DateTime dateFInicio = Convert.ToDateTime(dateFecI.Text);
            DateTime dateFFin = Convert.ToDateTime(dateFecF.Text);

            if (dateFInicio <= dateFFin)
            {
                wcfCRM.Service1Client objWcf = new wcfCRM.Service1Client();
                objWcf.Open();
```

```
                DataTable dtNotif = objWcf.getNotificacionesHist(codigoNotif,
dateFecI.Text, dateFecF.Text);
                objWcf.Close();
                if (dtNotif.Rows.Count > 0)
                {
                    storegrdNotif.DataSource = dtNotif;
                    storegrdNotif.DataBind();
                    X.Msg.Notify("Información.", "Reporte <br/><b>Desde:</b> " +
dateFInicio.ToShortDateString() + "<br/><b>Hasta:</b> " +
dateFFin.ToShortDateString()).Show();
                }
                else
                {
                    X.Msg.Notify("Información.", "Sin resultados").Show();
                    storegrdNotif.DataSource = new DataTable();
                    storegrdNotif.DataBind();
                }
            }
            else
                this.msgAlertas("Alerta...!!!", "La fecha inicio no puede ser mayor que
la fecha fin.", MessageBox.Icon.ERROR, btnGenerarReport);
        }
        else
            this.msgAlertas("Alerta...!!!", "Seleccione una opción de las
notificaciones.", MessageBox.Icon.WARNING, btnGenerarReport);
    }
    catch (Exception ex)
    {
        this.msgAlertas("Excepción...!!!", ex.Message, MessageBox.Icon.ERROR);
    }
}
```

**Exportar**

```
protected void btnExportarNotif_Click(object sender, EventArgs e)
{
    string json = GridData.Value.ToString();
    if (json.Length > 2)
    {
        string[] grid = json.Split(',');
        string json_ = "[";
        for (int i = 0; i < grid.Length; i++) {
            if ( (grid[i].Contains("codigo")))
                json_ += "{";
            else if ((grid[i].Contains("id")))
            {
                json_ = json_.Substring(0, json_.Length - 1);
                json_ += "},";
            }
            else
                json_ += grid[i] + ",";
        }
        json_ = json_.Substring(0, json_.Length - 1);
        json_ += "]";
        StoreSubmitDataEventArgs eSubmit = new StoreSubmitDataEventArgs(json_, null);
        XmlNode xml = eSubmit.Xml;
        this.Response.Clear();
        this.Response.ContentType = "application/ms-excel";
        this.Response.AddHeader("Content-Disposition", "attachment;
filename=RptNotificaciones.xls");
        this.Response.Charset = "utf-8";
        XslCompiledTransform xtExcel = new XslCompiledTransform();
        xtExcel.Load(Server.MapPath("Excel.xsl"));
```

```
        xtExcel.Transform(xml, null, this.Response.OutputStream);
        this.Response.End();
    }
    else
        this.msgAlertas("Mensaje...!!!", "No existen registros para generar el reporte
excel.", MessageBox.Icon.WARNING, btnExportarNotif);
}
```

## MÉTODOS MÓDULO ANÁLISIS

(frmCubo.aspx)

```
<form id="form1" runat="server">
<asp:ScriptManager ID="ScriptManager1" runat="server"></asp:ScriptManager>
<radarcube:TMDcube id="TMDCube1" runat="server"
ConnectionString="Provider=MSOLAP.5;Data Source=localhost;Catalog=CRMDWCube" />
<asp:UpdatePanel ID="UpdatePanel1" runat="server">
      ContentTemplate>
         <table width="100%">
             <tr>
                 <td>
                     <radarcube:TOLAPGridFilters ID="Filters1" runat="server"
GridID="TOLAPGrid1" BackColor="Transparent"
                         BorderColor="Transparent" />
                 </td>
             </tr>
             <tr>
                 <td>
                     <radarcube:TOLAPGrid ID="TOLAPGrid1" runat="server"
CubeID="TMDCube1" Width="95%" MaxTextLength="40" AllowPaging="True"
StructureTreeWidth="100%" DefaultLanguageCode="es" Height="500px" LinesInPage="30"
MaxRowsInGrid="5000000" MaxColumnsInGrid="5000" EmptyCellString="0" AllowScrolling="True"
Skin="Redmond" SupportEMail="carlostutillo@farmaenlace.com" CurrencyFormatString="$#,###"
ShowModificationAreas="false" ShowMeasuresPositionArea="false" AllowEditing="True"
HierarchiesDisplayMode="TableLike" Title="OLAP Grid"  >
                             <LegendSettings Background="White"  ShowLegends="false"  />
                             <HierarchyEditorStyle ItemsInPage="15" TreeHeight="300" />
                         </radarcube:TOLAPGrid
                     </td>
                 </tr>
             </table>
         </ContentTemplate>
     </asp:UpdatePanel>

     <radarcube:TOLAPExport ID="TOLAPExport1" runat="server" GridID="TOLAPGrid1"
IgnoreGridPaging="True">
         <Options>
             <CellMemberView AlingLine="Near" Font="Tahoma, 10pt" />
             <CellCaptionView AlingLine="Center" Font="Tahoma, 10pt, style=Bold" />
             <CellMemberTotalView AlingLine="Near" Font="Tahoma, 10pt" />
             <CellDataTotalView AlingLine="Center" Font="Tahoma, 10pt" />
             <CellDataView AlingLine="Center" Font="Tahoma, 10pt" />
             <CellGroupView AlingLine="Near" Font="Tahoma, 10pt" />
         </Options>
         <OptionsPDF>
             <PageNumber Font="Tahoma, 10pt" />
             <Header Font="Tahoma, 10pt" />
         </OptionsPDF>
     </radarcube:TOLAPExport>

</form>
```

```
// (frmCubo.aspx.cs)
protected void Page_Load(object sender, EventArgs e)
{
    try
    {
        if (Session["userLogin"] == null)
        {
            this.msgAlertas("Logout...!!!", "Sesión caducada o no iniciada.",
MessageBox.Icon.ERROR);
            return;
       }
       if (!IsPostBack)
       {
            Response.CacheControl = "no-cache";
            Response.AddHeader("Pragma", "no-cache");
            Response.Expires = -1;
            Skins skin = (Skins)Enum.Parse(typeof(Skins), "Redmond");
TOLAPGrid1.Skin = skin;
                    //SetModificationPanelsVisibility();

            TMDCube1.ConnectionString = "Provider=MSOLAP.5;Data
Source=localhost;Catalog=CRMDWCube";
            TMDCube1.CubeName = "CRMDW";
            TMDCube1.Active = true;
                    //+ set the measures visibility
              TOLAPGrid1.Measures.Find("[Measures].[Valor Neto]").Visible = true;
                                    }
    }
    catch (Exception ex)
    {
        X.Msg.Alert("Excepción...!!!",ex.Message).Show();
    }
}
```