**Sistema Artec**

**Resumen de actividades desarrolladas**

1. Lenguaje de programación:C#
2. Desarrollo en 5 capas: Entidades, BLL, DAL, GUI y Servicios
3. Acceso único a la Base de Datos a través de la capa de servicios
4. La clase de acceso a la BD es reutilizable y permite a otras capas iniciar y finalizar una transacción
5. Mapeador resuelve mediante reflection
6. Servicios:
   * Persistencia y recupero de datos
   * Backup y Restore BD
   * Idioma
   * Login implementando patrón Singleton
   * Encriptación y aplicación de hash
   * (Bitácora)
   * (Digitos Verificadores)
   * (Manejo de excepciones)
   * (Autorización mediante Familia Patente)
7. Interfaces y lógica programada:
   * Login (Autenticación)
   * Crear Solicitud
   * Backup y Restore
   * Crear Proveedor

Descartes:

26/06/2017

1. Las compras de software son únicamente de licencias permanentes, porque las que son por suscripción requieren una erogación de gastos continua, y eso conlleva a la elevación del pedido hacia la procuradora y el trámite de adquisición lo realiza el área de administración (se iría a una cuasi Licitación)
   * Eliminé fechasus y fehafinsusc de inventario
   * Eliminé IdTipoLicencia de Bien
   * Elimine relación Bien con Tipolicencia
   * Elimine Tabla TipoLicencia
2. Adquisicion
   * Elimine NroExpediente de Adquisicion
   * Eliminé NroOrdenCompra de Adquisicion
3. Tabla AsigDetalle
   * Eliminé TiempoAsignación de AsigDetalle porque no se contemplan los prestamos
4. Tabla TipoAsignación, la eliminé porque solo hay un tipo de asignación (ya que no se contemplan prestamos)
   * Eliminé IdTipoAsignacion en AsigDetalle
5. Idioma
   * Eliminé la columna IdFormulario de Etiqueta
   * Modifique la clave primaria en etiqueta quitando de la misma IdFormulario
   * Eliminé la tabla Formulario
6. Store [dbo].[CotizacionTraerPorSolicitud]
   * Modifiqué el select

Estaba: --select Coti.IdCotizacion, Coti.MontoCotizado, Coti.FechaCotizacion, Prov.IdProveedor, Prov.AliasProv, Coti.IdPartidaDetalle, Coti.IdPartida, Det.IdSolicitudDetalle, Det.IdSolicitud

Quedó: select Coti.IdCotizacion, Coti.MontoCotizado, Coti.FechaCotizacion, Prov.IdProveedor, Prov.AliasProv, Det.IdSolicitudDetalle, Det.IdSolicitud

1. Porque pinchaba cuando quería agregar una cotización a un detalle que tenía cero cotizaciones:
   * Agregue lo resaltado en frmsolicitudmodificar:

frmCotizaciones UnFrmCotizaciones = new frmCotizaciones(unaSolicitud.unosDetallesSolicitud[e.RowIndex].unasCotizaciones, unDetSolic);

* + Agregue lo resaltado (y quite lo comentado) en frmCotizaciones:

unaCotiz.unDetalleAsociado.IdSolicitud = unDetSolic.IdSolicitud;//unasCotizaciones[0].unDetalleAsociado.IdSolicitud;

unaCotiz.unDetalleAsociado.IdSolicitudDetalle = unDetSolic.IdSolicitudDetalle;//unasCotizaciones[0].unDetalleAsociado.IdSolicitudDetalle;

1. Se eliminó table TipoAdquisicion y referencia desde Adquisición
   * Modifique AdquisicionCrear en consecuencia
2. Agregue campo IdPartidaDetalle en tabla Inventario, para poder relacionar el inventario, con la compra y la solicitud
3. Agregar prop Bien en categoría para poder acceder al tipo de bien desde el detalle de solic, creo q ya no}
4. Eliminar IdAdquisicion de tabla PartidaDetalle (posiblemente no lo use nunca)
5. Quitar el agente relacionado al software
6. Quitar la tabla depósito, porque no hay varios depósitos, además se compra lo solicitado y se lo entrega, no se stockea, lo de consultar stock era para cuando se entrega en forma parcial algo

You need to create an event handler for the user control that is raised when an event from within the user control is fired. This will allow you to bubble the event up the chain so you can handle the event from the form.

When clicking Button1 on the UserControl, i'll fire Button1\_Click which triggers UserControl\_ButtonClick on the form:

User control:

public event EventHandler ButtonClick;

protected void Button1\_Click(object sender, EventArgs e)

{

//bubble the event up to the parent

if (this.ButtonClick!= null)

this.ButtonClick(this, e);

}

Form:

UserControl1.ButtonClick += new EventHandler(UserControl\_ButtonClick);

protected void UserControl\_ButtonClick(object sender, EventArgs e)

{

//handle the event

}

O sino::

***User Control***

public event EventHandler StatusUpdated;

private void FunctionThatRaisesEvent()

{

//Null check makes sure the main page is attached to the event

if (this.StatusUpdated != null)

this.StatusUpdated(this, new EventArgs());

}

***Main Page/Form***

public void MyApp()

{

//USERCONTROL = your control with the StatusUpdated event

this.USERCONTROL.StatusUpdated += new EventHandler(MyEventHandlerFunction\_StatusUpdated);

}

public void MyEventHandlerFunction\_StatusUpdated(object sender, EventArgs e)

{

//your code here

}