MANUAL TÉCNICO

DossierTec

Historial de Versiones

Versión	Fecha	Detalle	Autor
1	05/08/2021	Primera versión	PZDY

CONTENTS

1. ARQUITECTURA GENERAL

Arquitectura de componentes

La arquitectura de paquetes representa la organización general de los componentes de software.

Los componentes mantendrán una independencia entre ellos, tomando como referencia las listas de información.

Arquitectura de componentes

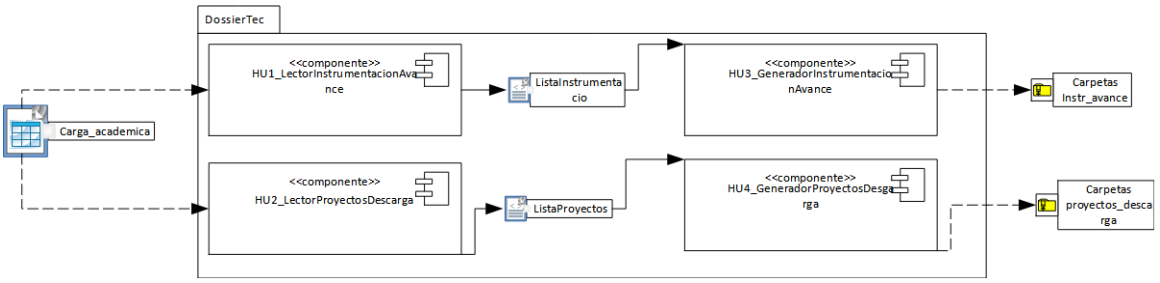


Tabla de componentes

Paquete	Componente	Descripción
DossierTec	LectorInstrumentacionAvance	El componente mostrará un dialogo para seleccionar el archivo de CargaAcademica, validará los datos y generará un listado con la carrera, docente, materia, semestre y grupo.
	LectorProyectosDescarga	El componente mostrará un dialogo para seleccionar el archivo de CargaAcademica, validará los datos y generará en un listado con docente y proyecto de descarga.
	GeneradorInstrumentacio-nAvance	El componente recibirá como entrada el listado de instrumentaciones y mostrará un dialogo para seleccionar la ubicación y generar una estructura del repositorio de instrumentación y avance (Carrera, docente y materia, el nombre de la materia debe estar concatenada con el semestre y grupo)

	GeneradorProyectosDesgarga	El componente recibirá como entrada el listado de proyectos de descarga y mostrará un dialogo para seleccionar la ubicación y generar una estructura del repositorio de proyectos de descarga (Docente y proyectos de descarga)
	ListasInformacion	El componente contendrá las estructuras de las listas de información usadas para estructurar los datos para generar las carpetas de los repositorios.

Diseño de base de datos

La información entre los módulos se intercambiará usando listas de información (representados con **List**) con la siguiente estructura:

Lista 1 datos para repositorio de instrumentación y avance

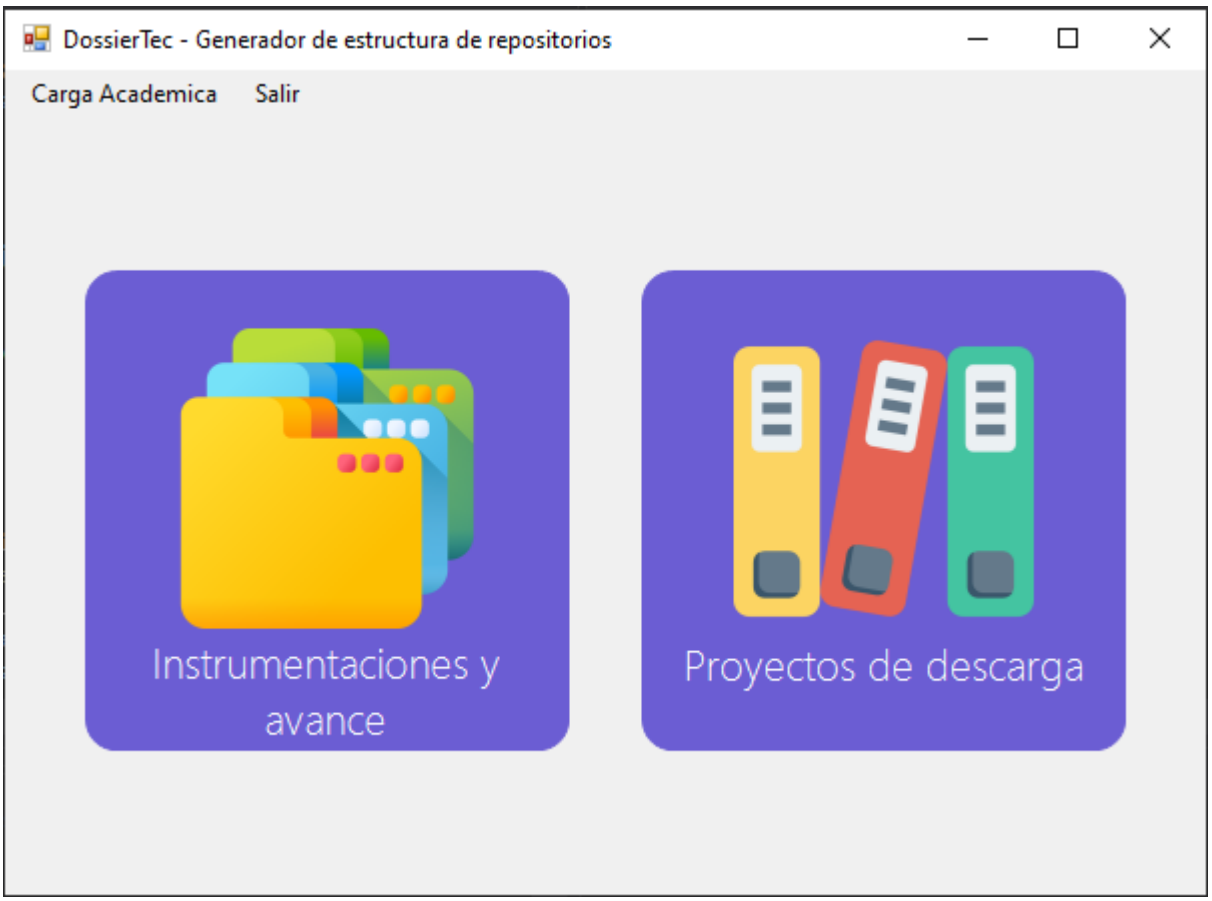
Carrera	Docente	Materia	Semestre	Grupo

Lista 2 datos para el repositorio de proyectos de descarga

Docente	ProyectoDescarga

2. PRINCIPAL

FrmMenu




```
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using LectorInstrumentacionAvance;
using LectorProyectosDescarga;

namespace DossierTec
{
    public partial class FrmMenu : Form
    {
        public FrmMenu()
        {
            InitializeComponent();
        }

        private void salirToolStripMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void toolStripButton1_Click(object sender, EventArgs e)
```

```
    {
        FrmSeleccionarInstrumentacion form = new FrmSeleccionarInstrumentacion();
        form.ShowDialog();
    }

    private void toolStripButton2_Click(object sender, EventArgs e)
    {
        FrmSeleccionarProyectos form = new FrmSeleccionarProyectos();
        form.ShowDialog();
    }

    private void instrumentaicionesYAvanceToolStripMenuItem_Click(object sender, EventArgs
e)
    {
        AbrirFormulario(new FrmSeleccionarInstrumentacion());
    }

    private void proyectosDeDescargaToolStripMenuItem_Click(object sender, EventArgs e)
    {
        AbrirFormulario(new FrmSeleccionarProyectos());
    }

    private void BtnInstrumentacion_Click(object sender, EventArgs e)
    {
        AbrirFormulario(new FrmSeleccionarInstrumentacion());
    }

    private void BtnProyectos_Click(object sender, EventArgs e)
    {
        AbrirFormulario(new FrmSeleccionarProyectos());
    }

    public Form FormActivo = null;
    public void AbrirFormulario(Form FormHijo)
    {
        if (FormActivo != null)
        {
            FormActivo.Close();
        }
        FormActivo = FormHijo;
        FormActivo.TopLevel = false;
        FormHijo.FormBorderStyle = FormBorderStyle.None;
        FormHijo.Dock = DockStyle.Fill;
        PnlPrincipal.Controls.Add(FormHijo);
        PnlPrincipal.Tag = FormHijo;
```

```
            FormHijo.BringToFront();
            FormHijo.Show();
        }
    }
}
```

3. LISTASINFORMACION

3.1. LISTAS.CS

```
using System;
using System.Collections.Generic;
using System.Text;

namespace ListasInformacion
{
    class Listas
    {
        /// <summary>
        /// Esta lista sirve para almacenar y validar los datos
        /// que se usarán para generar el repositorio de Instrumentaciones y Avance.
        /// </summary>
        static public List<Instrumentacion> ListaInstrumentaciones { get; set; }
        /// <summary>
        /// Esta lista sirve para almacenar y validar los datos
        /// que se usarán para generar el repositorio de Proyectos.
        /// </summary>
        static public List<Proyecto> ListaProyectos { get; set; }
    }
}
```

3.2. INSTRUMENTACION.CS

```
using System;
using System.Collections.Generic;
using System.Text;

namespace ListasInformacion
```

```
{
   class Instrumentacion
   {
      public Instrumentacion(string carrera, string docente, string materia, string semestre, string
grupo)
      {
         Carrera = carrera;
         Docente = docente;
         Materia = materia;
         Semestre = semestre;
         Grupo = grupo;
      }

      public String Carrera { get; set; }
      public String Docente { get; set; }
      public String Materia { get; set; }
      public String Semestre { get; set; }
      public String Grupo { get; set; }

      public override String ToString()
      {
         return Carrera + " - " + Docente + " - " + Materia + " - " + Semestre + " - " + Grupo;
         ///ISC-DanielArredondoSalcedo-AMD6A
      }
   }
}
```

3.3. PROYECTO.CS

```
using System;
using System.Collections.Generic;
using System.Text;

namespace ListasInformacion
{
   class Proyecto
   {
      public Proyecto()
      {
         Docente = "";
         ProyectoDescarga = "";
      }

      public Proyecto(string docente, string proyectoDescarga)
```

```
        {
            Docente = docente;
            ProyectoDescarga = proyectoDescarga;
        }

        public String Docente { get; set; }
        public String ProyectoDescarga { get; set; }

        public override String ToString()
        {
            return Docente + " - " + ProyectoDescarga;
        }
    }
}
```

4. INSTRUMENTACIÓN Y AVANCE

4.1. LECTOR INSTRUMENTACIÓN-AVANCE

Diseño de Clases

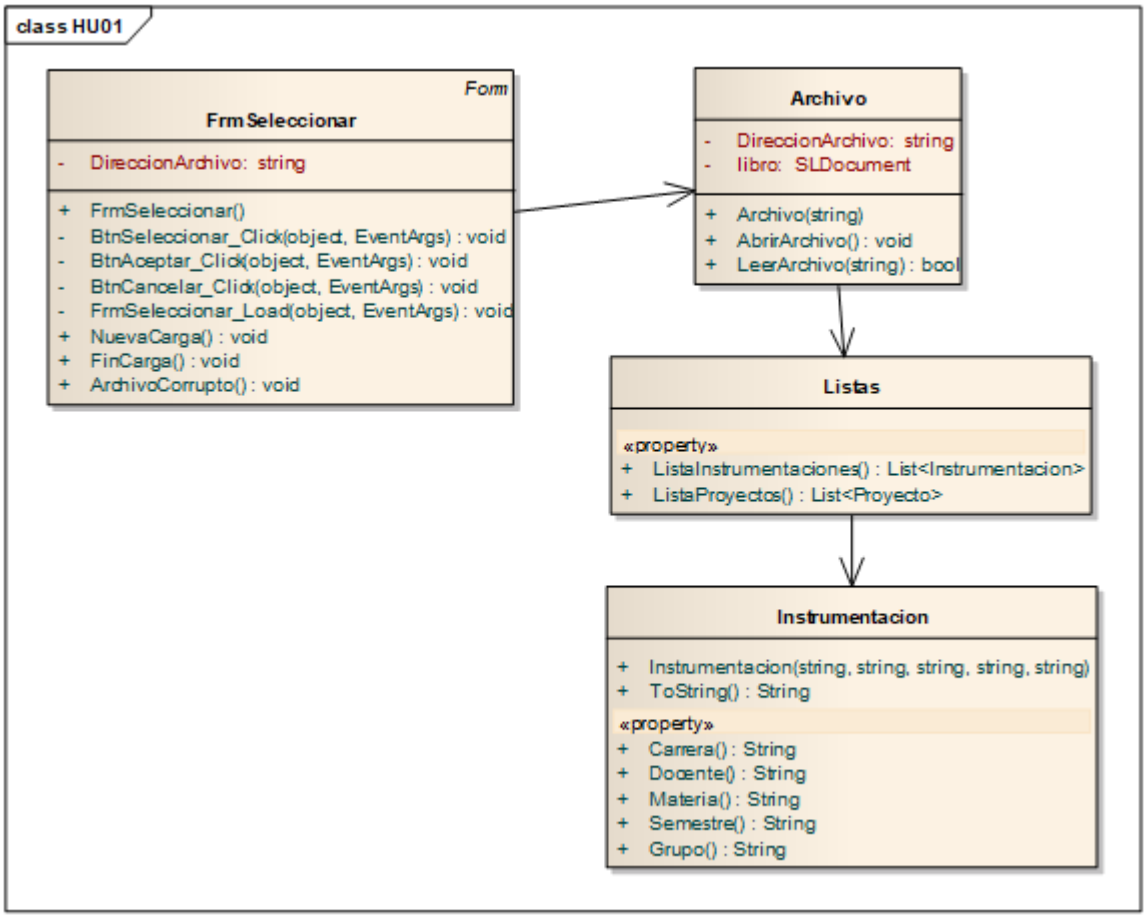
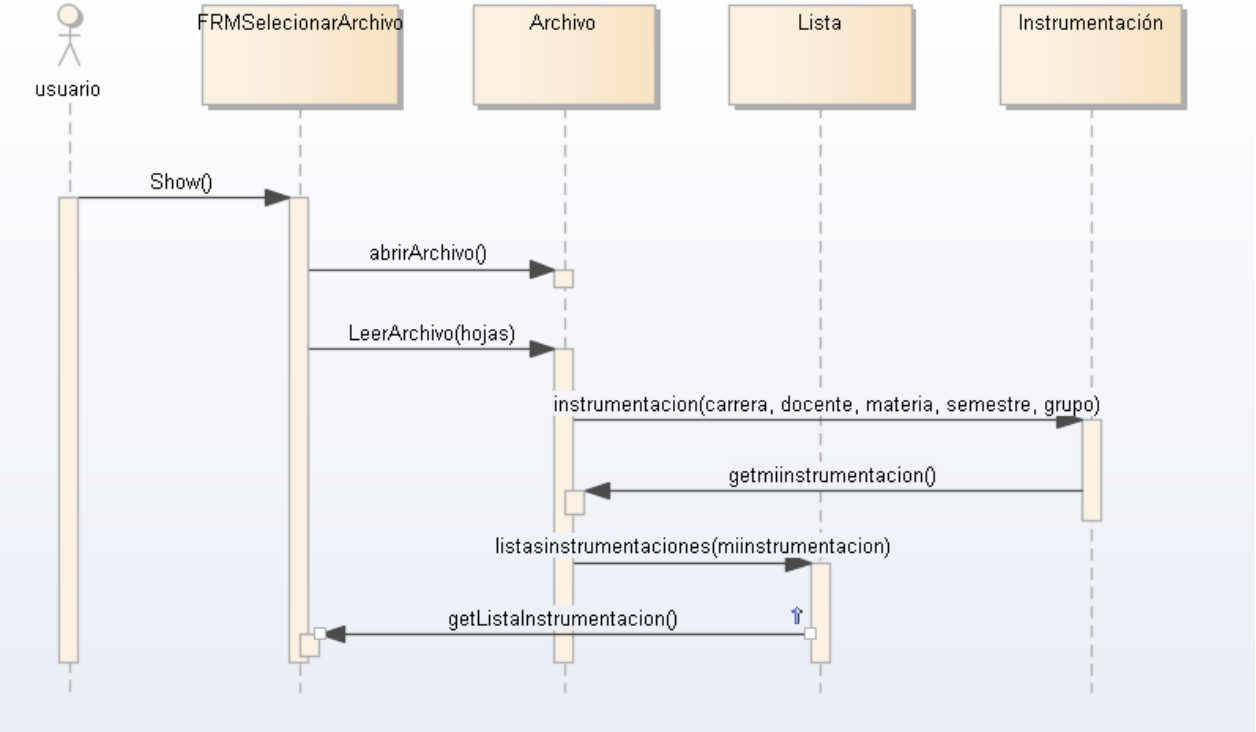
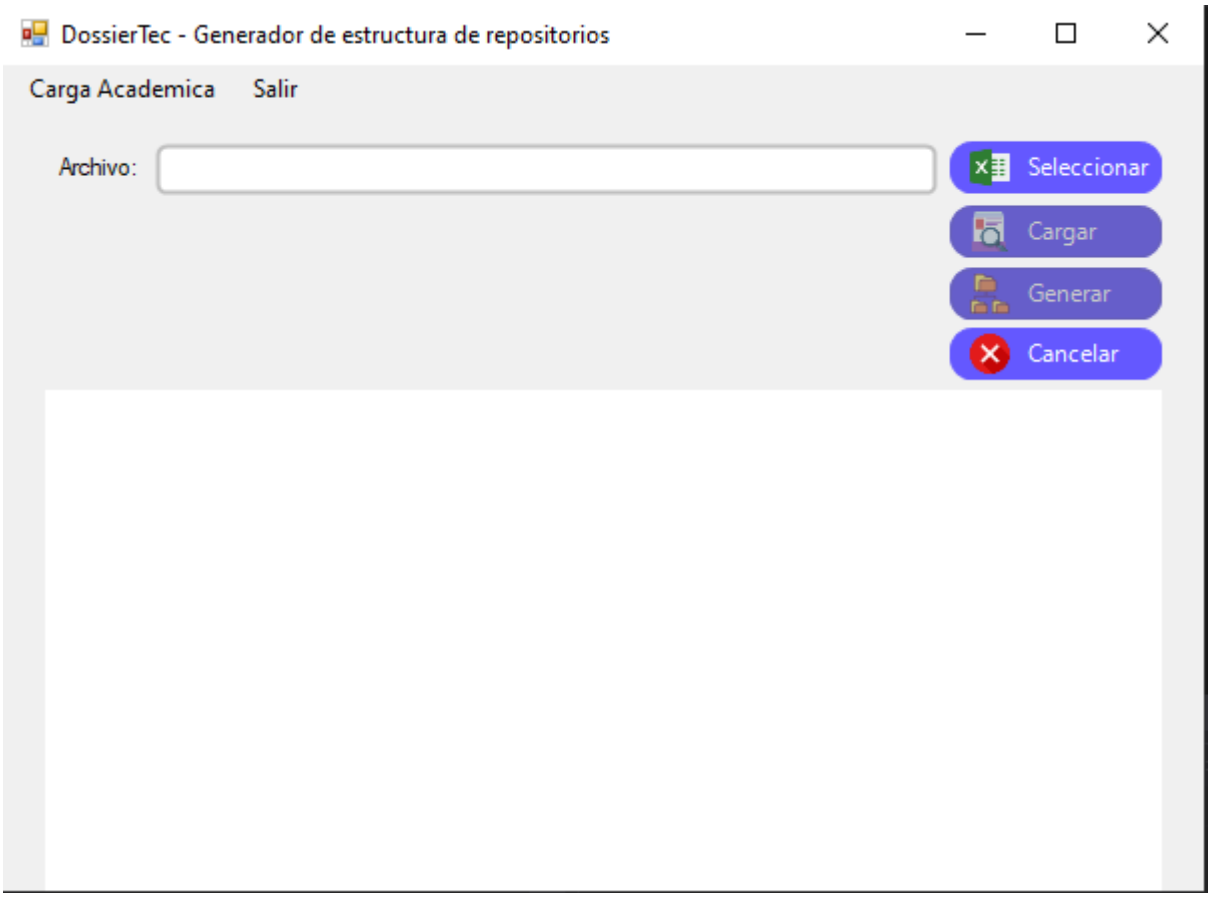


Diagrama de Secuencia



4.1.1. FRMSELECCIONARINSTRUMENTACION.CS




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Diagnostics;
using SpreadsheetLight;
using ListasInformacion;
using GeneradorInstrumentacionAvance;

namespace LectorInstrumentacionAvance
{
```

```
public partial class FrmSeleccionarInstrumentacion : Form
{
    string DireccionArchivo; // Para almacenar la direccion del archivo seleccionado
    public FrmSeleccionarInstrumentacion()
    {
        InitializeComponent();
        DireccionArchivo = "";
    }



    private void FrmSeleccionar_Load(object sender, EventArgs e)
    {
        BtnCargar.Enabled = false;
        BtnGenerar.Enabled = false;
        PbCarga.Visible = false;
        lblCargando.Text = "";
        TxtArchivo.Enabled = false;

    }


    private void BtnSeleccionar_Click(object sender, EventArgs e)
    {
        OpenFileDialog AbrirArchivo = new OpenFileDialog();
        AbrirArchivo.Title = "Seleccionar un Archivo";
        AbrirArchivo.Filter = "Archivos de Excel (*.xls;*.xlsx)|*.xls;*.xlsx"; // Para evitar que se selec-
cionen archivos que no sean xls o xlsx
        AbrirArchivo.FileName = this.TxtArchivo.Text;



        if (AbrirArchivo.ShowDialog() == System.Windows.Forms.DialogResult.OK)
        {
            this.TxtArchivo.Text = AbrirArchivo.FileName;
            this.DireccionArchivo = TxtArchivo.Text;
        }
        if (TxtArchivo.Text == "")
        {
            BtnCargar.Enabled = false;
        }
        else
        {
            BtnCargar.Enabled = true;
            BtnGenerar.Enabled = false;
```

```
        }
    }

    private void BtnAceptar_Click(object sender, EventArgs e)
    {
        NuevaCarga();
        Archivo mArchivo = new Archivo(DireccionArchivo);
        if (mArchivo.AbrirArchivo())
        {

            float porcentaje = 0;
            SLDocument libro;
            libro = new SLDocument(DireccionArchivo);
            string[] hojas = libro.GetWorksheetNames().ToArray();
            PbCarga.Maximum = hojas.Length;
            float carga = 100 / (hojas.Length);
            for (int i = 0; i < hojas.Length; i++)
            {
                if (hojas[i] == "Docentes") continue;
                porcentaje = porcentaje + carga;
                BtnSeleccionar.Enabled = false;

                if (mArchivo.LeerArchivo(hojas[i]))
                {
                    lblCargando.Text = "cargando.." + porcentaje + "%";
                    this.Text = "cargando... " + porcentaje + "%";
                    lblHojas.Text = "Hojas leidas: " + i;
                    PbCarga.PerformStep();
                }
            }
            FinCarga();
        }
        else
        {
            ArchivoCorrupto();
        }
    }

    private void BtnCancelar_Click_1(object sender, EventArgs e)
    {
        this.Close();
    }

    public void NuevaCarga()
    {
```

```
            this.Text = "Cargando...";
            lblCargando.Text = "Cargando...";
            lblCargando.Visible = true;
            PbCarga.Value = 0;
            PbCarga.Visible = true;
            BtnSeleccionar.Enabled = false;
            BtnCargar.Enabled = false;
            BtnGenerar.Enabled = false;
        }
        public void FinCarga()
        {
            this.Text = "Selecionar Archivo";
            PbCarga.Value = PbCarga.Maximum;
            BtnCargar.Enabled = false;
            BtnSeleccionar.Enabled = true;
            BtnGenerar.Enabled = true;
            DgvLista.DataSource = Listas.ListaInstrumentaciones;
            lblCargando.Text = "Carga Completada";
            BtnCancelar.Enabled = true;
        }
        public void ArchivoCorrupto()
        {
            lblCargando.Text = "Error en el archivo";
            PbCarga.Value = PbCarga.Maximum;
            lblHojas.Text = "Hojas leidas: 0";
            BtnSeleccionar.Enabled = true;
            this.Text = "Selecionar Archivo";
        }

        private void BtnGenerar_Click(object sender, EventArgs e)
        {
            //Mostrar el formulario para seleccionar la ubicacion
            FrmSeleccionarUbicacion mFrmSeleccionar = new FrmSeleccionarUbicacion();
            mFrmSeleccionar.ShowDialog();
        }
    }
}
```

4.1.2. ARCHIVO.CS

```
using System;
using System.Collections.Generic;
using System.Text;
```

```
using System.Diagnostics;
using System.Windows.Forms;
using SpreadsheetLight;
using ListasInformacion;

namespace LectorInstrumentacionAvance
{
   public class Archivo
   {

      string DireccionArchivo;
      SLDocument libro;
      public Archivo(string DireccionArchivo)
      {
         this.DireccionArchivo = DireccionArchivo;
         Listas.ListaInstrumentaciones = new List<Instrumentacion>();

      }

      public bool AbrirArchivo()
      {

         try
         {
            libro = new SLDocument(DireccionArchivo);
            return true;
         }
         catch (Exception ex)
         {
            MessageBox.Show("Error al abrir el archivo, verifique que sea un archivo válido o no
esté siendo usado por otra aplicación ");
            return false;
         }

      }

      //mtodo para buscar los datos en las hojas de excel y llenar la lista
      public bool LeerArchivo(string Hoja)
      {
         // try por si existe algun error en la hoja
         try
         {

            libro.SelectWorksheet(Hoja);
```

```
            //por si no existe la hoja omitirla
            if (libro.SelectWorksheet(Hoja) == true)
            {
                int ColumnaDocente = 4;
                int ColumnaSemestre = 2;
                int ColumnaGrupo = 3;
                int ColumnaMateria = 1;
                string Docente = "";
                string Semestre = "";
                string Grupo = "";
                string Materia = "";
                string Carrera = Hoja;
                Instrumentacion mInstrumentacion;
                for (int Fila = 2; Fila <= 100; Fila++)
                {
                    if (!string.IsNullOrEmpty(libro.GetCellValueAsString(Fila, ColumnaMateria)))
                    {
                        Materia = libro.GetCellValueAsString(Fila, ColumnaMateria);
                    }
                    if (!string.IsNullOrEmpty(libro.GetCellValueAsString(Fila, ColumnaSemestre)))
                    {
                        Semestre = libro.GetCellValueAsString(Fila, ColumnaSemestre);
                    }
                    if (!string.IsNullOrEmpty(libro.GetCellValueAsString(Fila, ColumnaGrupo)))
                    {
                        Grupo = libro.GetCellValueAsString(Fila, ColumnaGrupo);
                    }
                    if (!string.IsNullOrEmpty(libro.GetCellValueAsString(Fila, ColumnaDocente)))
                    {
                        Docente = libro.GetCellValueAsString(Fila, ColumnaDocente);
                    }
                    if (libro.GetCellValueAsString(Fila, ColumnaMateria) == "" &&
                       libro.GetCellValueAsString(Fila + 1, ColumnaMateria) == "" &&
                       libro.GetCellValueAsString(Fila - 1, ColumnaMateria) == "")
                    {
                        break;
                    }
                    if (Carrera != "" && Docente != "" && Materia != "" && Semestre != "" && Grupo
!= "")
                    {
                        mInstrumentacion = new Instrumentacion(Carrera, Docente, Materia, Semestre,
Grupo);
                        Listas.ListaInstrumentaciones.Add(mInstrumentacion);
```

```
                }
                Docente = "";
                Semestre = "";
                Grupo = "";
                Materia = "";
            }
        }
        else
        {
            MessageBox.Show("¡No fue posible encontrar la Hoja " + Hoja + "!");
        }
        return true;
    }
    //por si existe un error en alguna hoja
    catch (Exception ex)
    {
        MessageBox.Show("¡Error al leer el archivo ! " + ex.ToString());
        return false;
    }


  }


}
```

4.2. GENERADOR INSTRUMENTACIÓN-AVANCE

Diseño de Clases

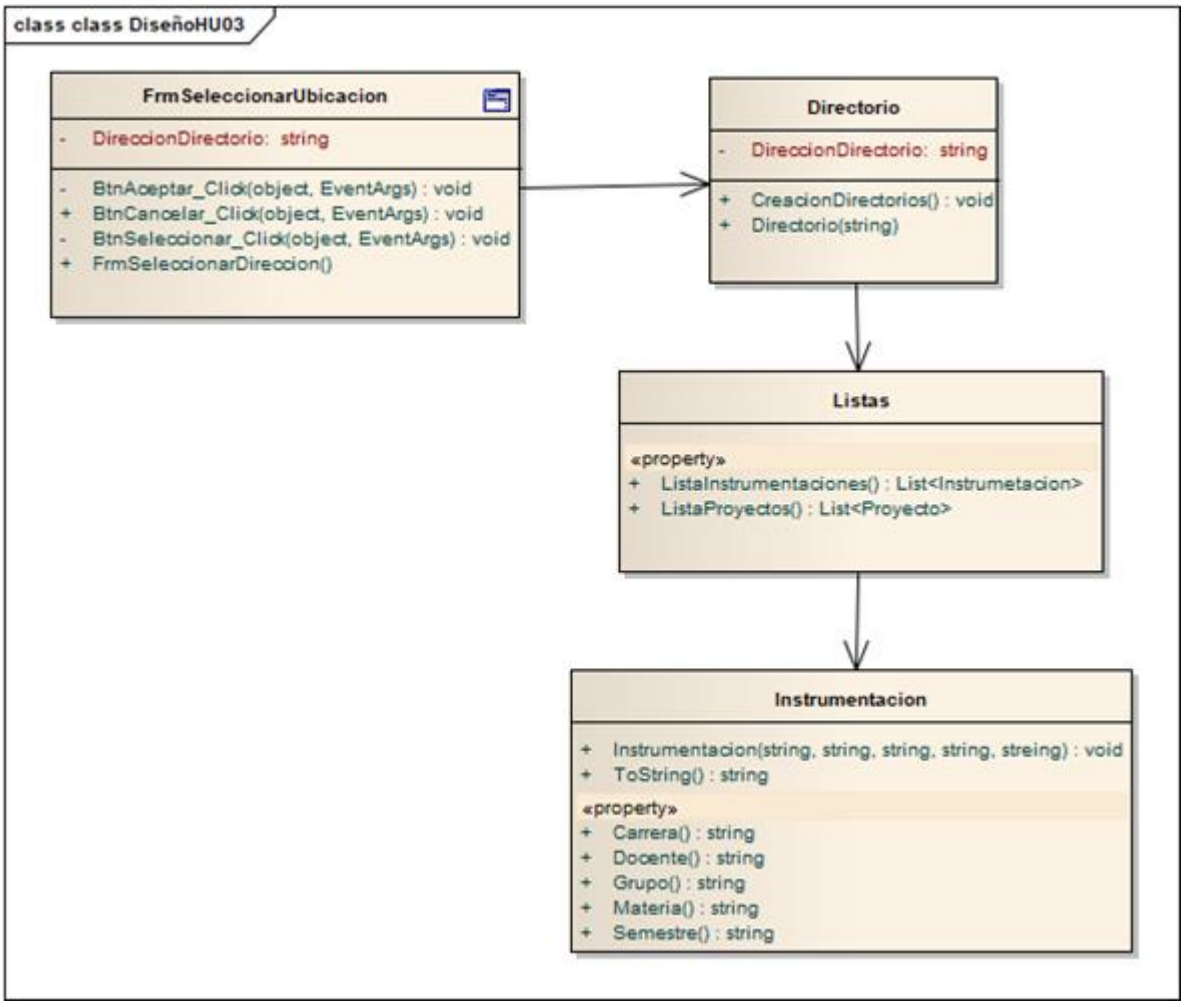
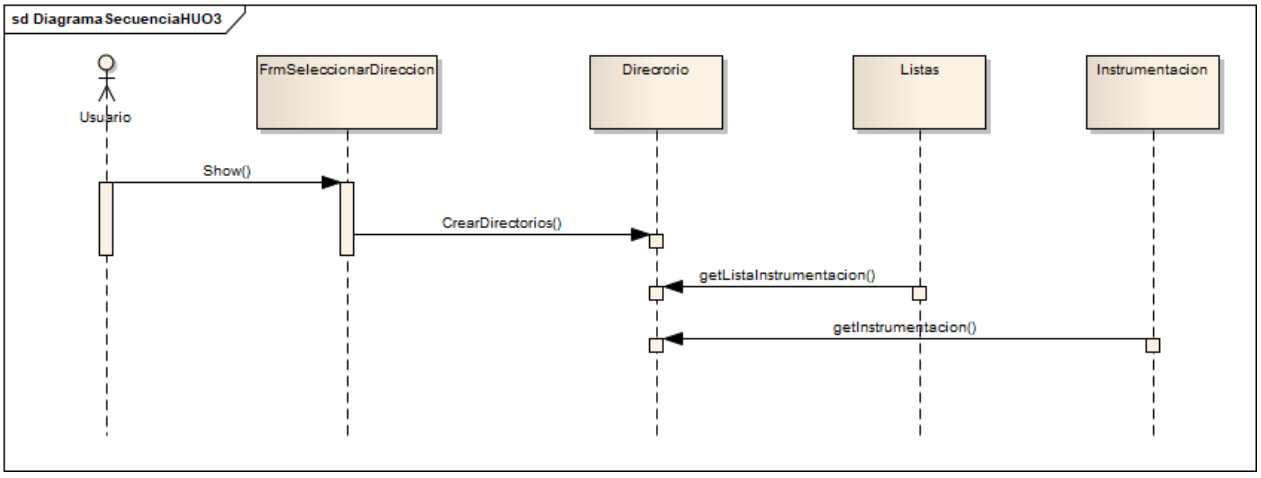
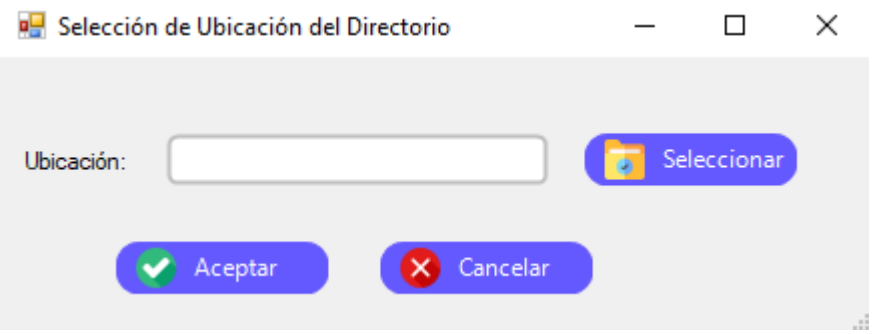


Diagrama de Secuencia



4.2.1. FRMSELECCIONARUBICACION.CS




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Collections;
using System.IO;

namespace GeneradorInstrumentacionAvance
{
    public partial class FrmSeleccionarUbicacion : Form
    {
        string DireccionDirectorio;
        public FrmSeleccionarUbicacion()
        {
            InitializeComponent();
            DireccionDirectorio = "";

        }


        private void BtnCancelar_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void BtnAceptar_Click(object sender, EventArgs e)
        {
            //Metodo para Mandar llamar a la Clase Directorio y crearlos
            if (TxtDireccion.Text == "")
            {
```

```
            MessageBox.Show("No se ha Seleccionado una Ubicación");
        }
        else
        {
            if (Directory.Exists(TxtDireccion.Text))
            {
                Directorio mDirectorio = new Directorio(this.DireccionDirectorio);

                //Ejemplo para Probar
                // this.llenarLista();

                mDirectorio.CreacionDirectorios();
                MessageBox.Show("El Directorio ha sido Creado");
                this.Close();
            }
            else
            {
                MessageBox.Show("{0} No es un Directorio Válido", TxtDireccion.Text);
            }
        }
    }

    private void BtnSeleccionar_Click(object sender, EventArgs e)
    {
        // Para seleccionar la Ubicación del Directorio
        FolderBrowserDialog Ubicacion = new FolderBrowserDialog();
        if (Ubicacion.ShowDialog() == DialogResult.OK)
        {
            TxtDireccion.Text = Ubicacion.SelectedPath;
            this.DireccionDirectorio = TxtDireccion.Text;
        }
        else
        {
            MessageBox.Show("La ruta es Inaccesible");
        }


        if (TxtDireccion.Text == "")
        {
            BtnAceptar.Enabled = false;
        }
        else
        {
            BtnAceptar.Enabled = true;
        }
```

```
        }

        //Ejemplo para Probar con una Lista
        /*private void llenarLista() {
            Archivo mArchivo = new Archivo();
            string[] hojas = { "ISC", "TICS", "IA", "IGE", "IEM", "CP", "IIA", "CP_S1", "CP_S2", "IA_S1",
"IA_S2", "EaD_IA_S1",
                "EaD_IA_S2", "EaD_CP_S1", "EaD_CP_S2", "EaD_ISC_S1", "EaD_ISC_S2" };
            for (int i = 0; i < hojas.Length; i++)
            {
                mArchivo.LeerArchivo(hojas[i]);
            }

        }*/
    }
}
```

4.2.2. DIRECTORIO.CS

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;
using System.Windows.Forms;
using ListasInformacion;

namespace GeneradorInstrumentacionAvance
{
    public class Directorio
    {
        string DireccionDirectorio; //Ubicación de nonde estará el directorio

        public Directorio(string Direccion) {
            this.DireccionDirectorio = Direccion;
        }

        public void CreacionDirectorios() {
            //Método para recorer la Lista y creando los directorios
            Listas.ListaInstrumentaciones = Listas.ListaInstrumentaciones;
            string Carrera = "";
            string Docente = "";
            string Materia = "";
            string Semestre = "";
            string Grupo = "";
```

```
        string DirExistentes = "";
        foreach (Instrumentacion mInstrumentacion in Listas.ListaInstrumentaciones){

            Carrera = mInstrumentacion.Carrera;
            Docente = mInstrumentacion.Docente;
            Materia = mInstrumentacion.Materia;
            Semestre = mInstrumentacion.Semestre;
            Grupo = mInstrumentacion.Grupo;
            // Especificamos la ruta completa del directorio a crear.
            string path = @""+this.DireccionDirectorio +"\\Instr-Avance"+ "\\"+Carrera + "\\"+Do-
cente+"\\"+Materia+"_"+Semestre + Grupo + " ";
            //MessageBox.Show(path);

            try {
               // Verifica si el directorio existe.
               if (Directory.Exists(path)){
                  DirExistentes= DirExistentes+" "+ Carrera + "\\" + Docente + "\\" + Materia + "_"
+ Semestre + Grupo + "\n";
               }else{
                  //Crea la ruta completa de directorios, junto con sus subcarpetas
                  DirectoryInfo di = Directory.CreateDirectory(path);
                  Console.WriteLine("El directorio ha sido creado {0}.", Directory.GetCreation-
Time(path));
               }
            }catch (Exception ex){
               MessageBox.Show("Error: {0}", ex.ToString());
            }

        }
        if(!DirExistentes.Equals(""))
            MessageBox.Show("Los siguientes directorios ya existian: \n "+ DirExistentes.Subs-
tring(0,1000) +"...");

      }
   }
}
```

5. PROYECTOS DE DESCARGA

5.1. LECTOR PROYECTOS DE DESCARGA

Diseño de Clases

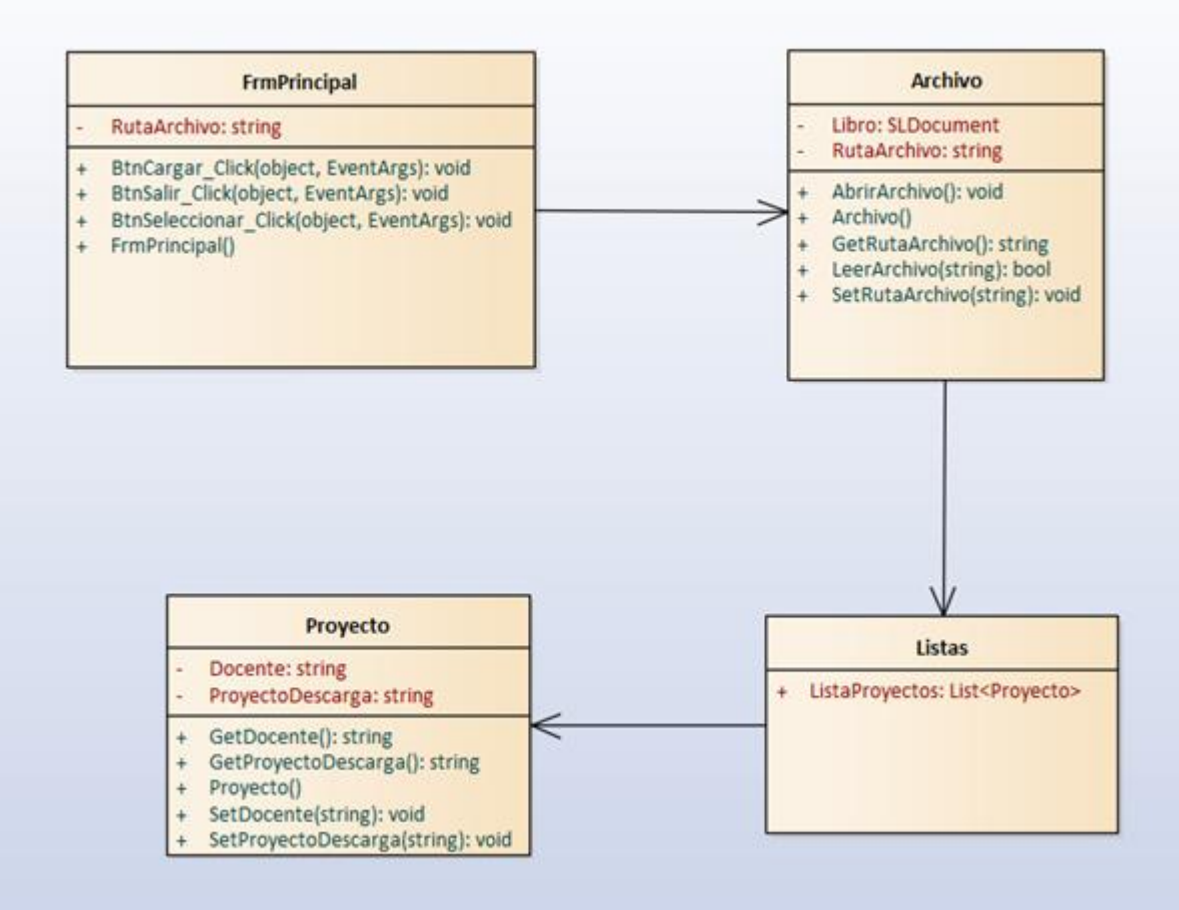
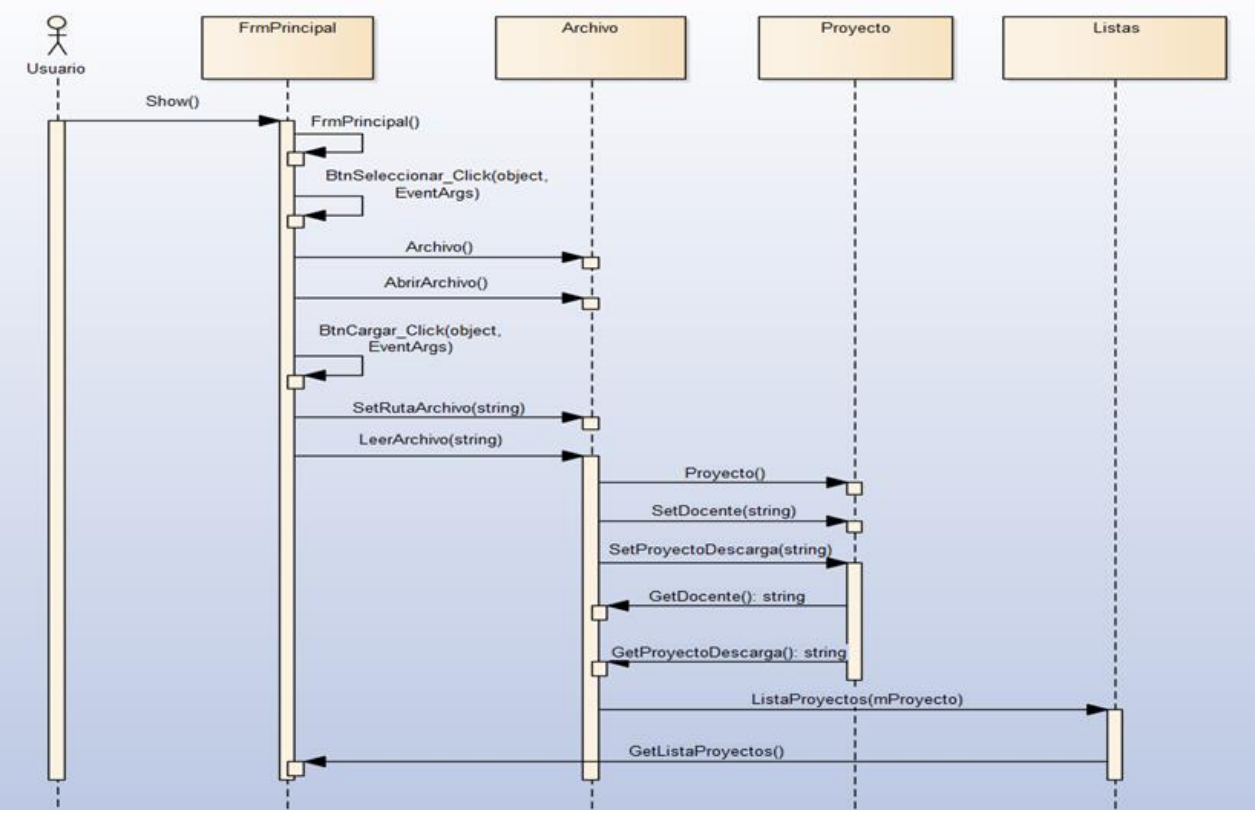
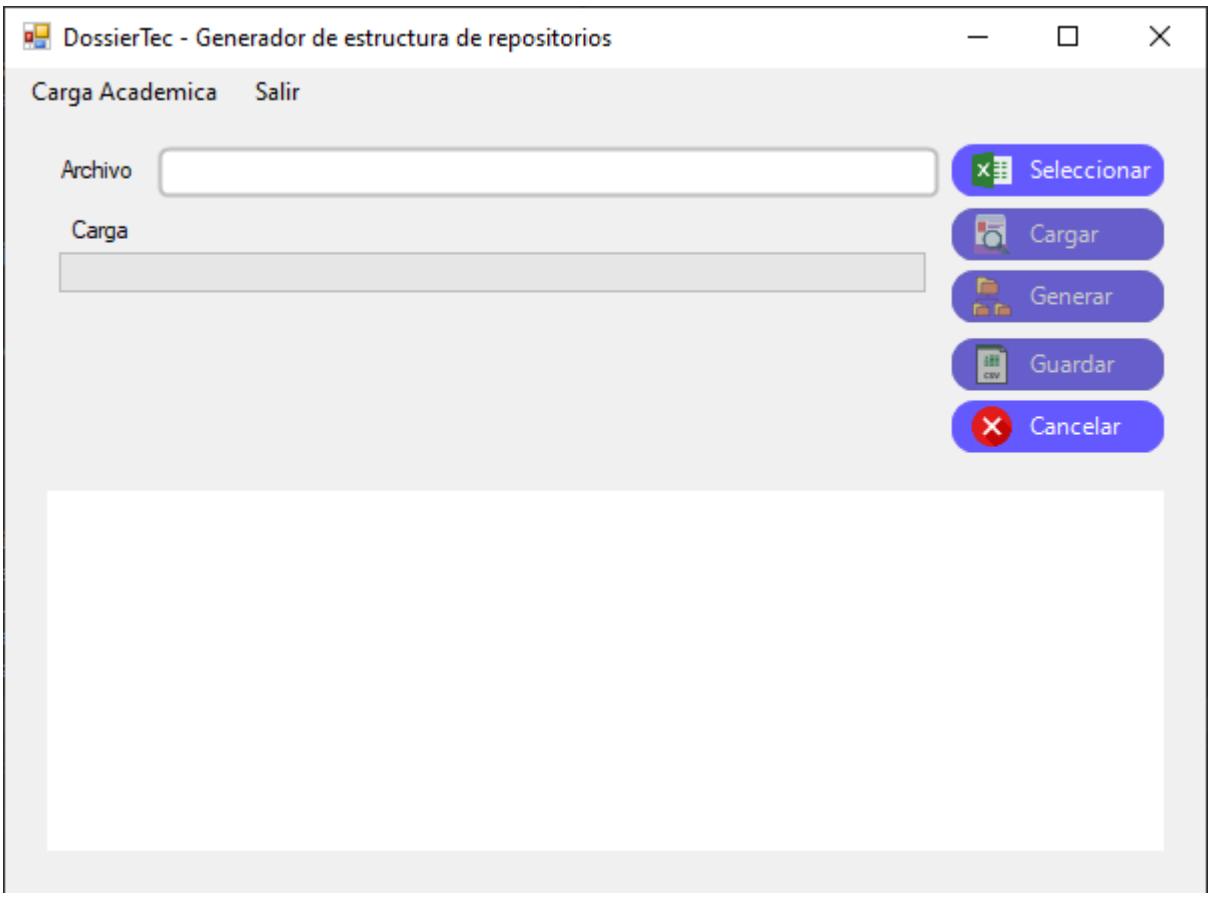


Diagrama de Secuencia



5.1.1. FRMSELECCIONARPROYECTOS.CS




```
using GeneradorProyectosDesgarga;
using GuardarCSV;
using ListasInformacion;
using SpreadsheetLight;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace LectorProyectosDescarga
{
    public partial class FrmSeleccionarProyectos : Form
    {
        private string RutaArchivo;
        private Archivo mArchivo;
        public FrmSeleccionarProyectos()
```

```
        {
            InitializeComponent();
            BtnCargar.Enabled = false;
            BtnGenerar.Enabled = false;
            BtnGuardarCSV.Enabled = false;
            TxtArchivo.Enabled = false;
        }



        private void BtnSeleccionar_Click(object sender, EventArgs e)
        {
            OpenFileDialog AbrirArchivo = new OpenFileDialog();
            AbrirArchivo.Title = "Seleccionar un Archivo";
            AbrirArchivo.Filter = "Archivos de Excel (*.xls;*.xlsx)|*.xls;*.xlsx"; // Para evitar que se selec-
cionen archivos que no sean xls o xlsx
            AbrirArchivo.FileName = this.TxtArchivo.Text;

            if (AbrirArchivo.ShowDialog() == System.Windows.Forms.DialogResult.OK)
            {
                this.TxtArchivo.Text = AbrirArchivo.FileName;
                this.RutaArchivo = TxtArchivo.Text;
            }
            if (TxtArchivo.Text == "")
            {
                BtnCargar.Enabled = false;
            }
            else
            {
                BtnCargar.Enabled = true;
                BtnGenerar.Enabled = false;
                BtnGuardarCSV.Enabled = false;

            }
        }

        private void BtnCargar_Click(object sender, EventArgs e)
        {
            mArchivo = new Archivo();
            mArchivo.padre = this;
            mArchivo.RutaArchivo = this.RutaArchivo;
            if (mArchivo.AbrirArchivo())
            {
                mArchivo.LeerArchivo("Docentes");
                DtgLista.DataSource = Listas.ListaProyectos;
```

```
            DtgLista.AutoResizeColumns();
            BtnCargar.Enabled = false;
            BtnGenerar.Enabled = true;
            BtnGuardarCSV.Enabled = true;
            PbCarga.Value = PbCarga.Maximum;
        }
    }

    private void BtnCancelar_Click(object sender, EventArgs e)
    {
        this.Close();
    }

   public void MostrarAvanceCarga()
    {
        PbCarga.PerformStep();
    }

    private void BtnGenerar_Click(object sender, EventArgs e)
    {
        //Mostrar el formulario para seleccionar la ubicacion
        FrmSeleccionarUbicacion_PD mFrmSeleccionar = new FrmSeleccionarUbicacion_PD();
        mFrmSeleccionar.ShowDialog();
    }

    private void BtnGuardarCSV_Click(object sender, EventArgs e)
    {
        FrmGenerarArchivoCsv mFrmGenerarArchivoCsv = new FrmGenerarArchivoCsv();
        mFrmGenerarArchivoCsv.ShowDialog();
    }
  }
}
```

5.1.2. ARCHIVO.CS

```
using ListasInformacion;
using SpreadsheetLight;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace LectorProyectosDescarga
```

```
{
    class Archivo
    {
        public string RutaArchivo { get; set; }
        SLDocument Libro;
        public FrmSeleccionarProyectos padre { get; set; }

        public Archivo(){
            RutaArchivo = "";
            Listas.ListaProyectos = new List<Proyecto>();
        }

        public bool AbrirArchivo() {

            try {
                Libro = new SLDocument(RutaArchivo);
                return true;
            }
            catch (Exception ex) {
                MessageBox.Show("Error al abrir el archivo, verifique que sea un archivo válido o no
esté siendo usado por otra aplicación ");
                return false;
            }
        }

        public bool LeerArchivo(string Hoja)
        {
            // try por si existe algun error en la hoja
            try {
                //por si no existe la hoja omitirla
                if (Libro.SelectWorksheet(Hoja) == true) {
                    string Docente = "";
                    string ProyectoDescarga="";
                    Proyecto mProyecto;
                    int Inicio=0;
                    int Fin = 0;
                    int j = 1;
                    while (true) {
                        if (Libro.GetCellValueAsString(1,j).ToUpper().Trim() == "PROYECTOS") {
                            Inicio = j;
                        }
                        if (Libro.GetCellValueAsString(2, j).ToUpper().Trim() == "CLASES") {
                            Fin = j;
                            if (Inicio == 0) {
                                break;
```

```
                }
            }
            if(Inicio!= 0 && Fin != 0) {
                break;
            }
            j++;
        }
        for (int i = 4; Libro.GetCellValueAsString(i,1).ToUpper().Trim() != "TOTALES"; i++){
            if (!string.IsNullOrEmpty(Libro.GetCellValueAsString(i,1))) {
                Docente = Libro.GetCellValueAsString(i, 1);
                for (int k = Inicio; k < Fin; k = k + 2) {
                    int NumeroUno;
                    int NumeroDos;
                    if (string.IsNullOrEmpty(Libro.GetCellValueAsString(i, k))) {
                        NumeroUno = 0;
                    } else {
                        NumeroUno = int.Parse(Libro.GetCellValueAsString(i, k));
                    }
                    if (string.IsNullOrEmpty(Libro.GetCellValueAsString(i, k + 1))) {
                        NumeroDos = 0;
                    } else {
                        NumeroDos = int.Parse(Libro.GetCellValueAsString(i, k + 1));
                    }

                    if ((NumeroUno + NumeroDos) > 0) {
                        if (!string.IsNullOrEmpty(Libro.GetCellValueAsString(2,k))) {
                            ProyectoDescarga = Libro.GetCellValueAsString(2, k);
                            if (Docente != "" && ProyectoDescarga != "") {
                                mProyecto = new Proyecto();
                                mProyecto.Docente = Docente;
                                mProyecto.ProyectoDescarga = ProyectoDescarga;
                                Listas.ListaProyectos.Add(mProyecto);
                            }
                        }
                    }
                }
            }
            padre.MostrarAvanceCarga();
        }
        Docente = "";
        ProyectoDescarga = "";
    } else {
        MessageBox.Show("¡No fue posible encontrar la Hoja " + Hoja + "!");
    }
    return true;
```

```
            }
            //por si existe un error en alguna hoja
            catch (Exception ex) {
                MessageBox.Show("¡Error al leer el archivo!\nPor favor verifique el formato.");
                return false;
            }
        }
    }
}
```

5.1.3. GUARDARCSV

Diseño de Clases

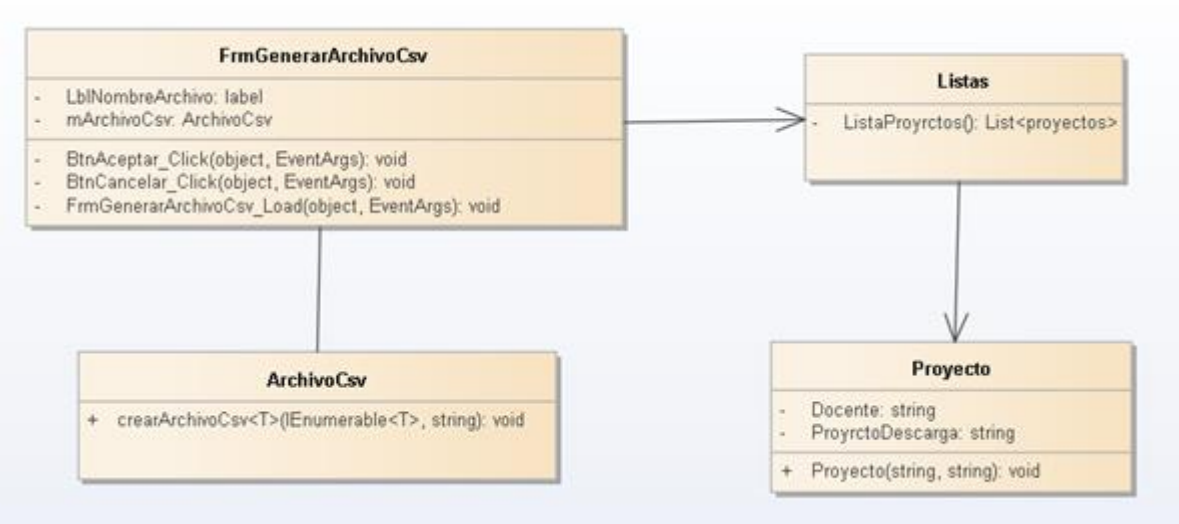
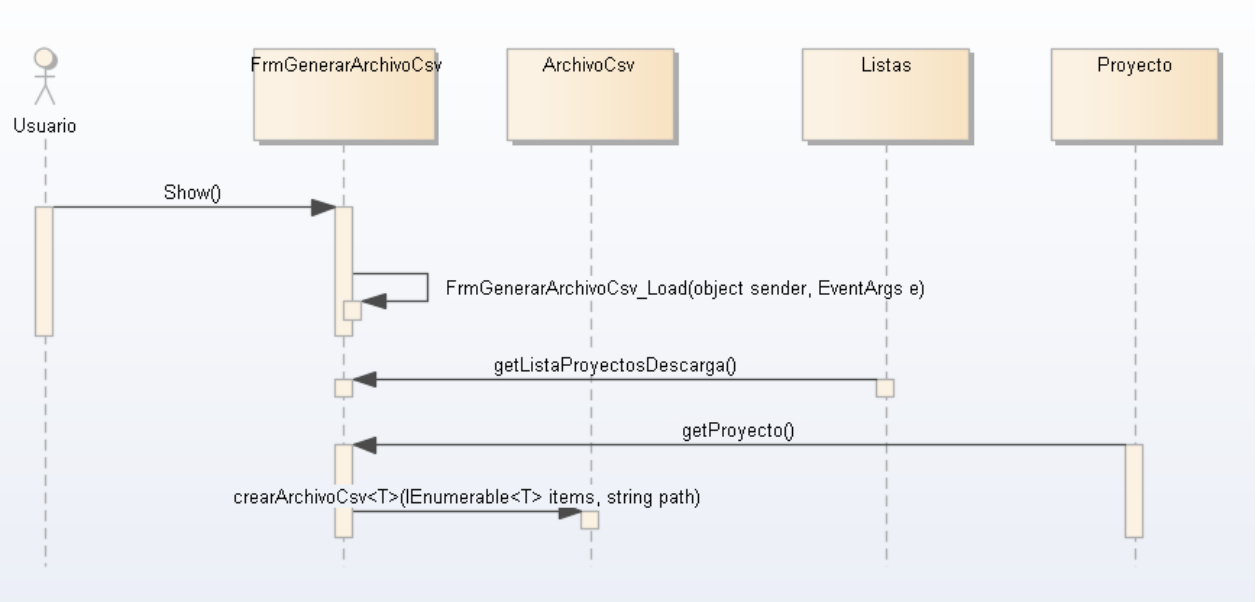
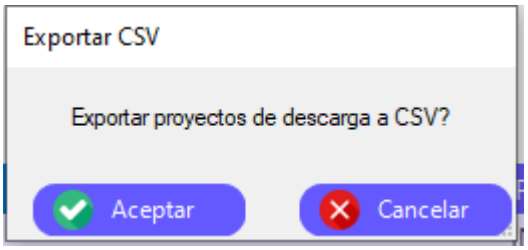


Diagrama de Secuencia



5.1.3.1. FRMEXPORTARCSV.CS




```
using ListasInformacion;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GuardarCSV
{
    public partial class FrmGenerarArchivoCsv : Form
    {
        private ArchivoCsv mArchivoCsv;
        public FrmGenerarArchivoCsv()
        {
            InitializeComponent();
            mArchivoCsv = new ArchivoCsv();
        }

        private void FrmGenerarArchivoCsv_Load(object sender, EventArgs e)
        {

        }

        private void BtnAceptar_Click(object sender, EventArgs e)
        {
            FolderBrowserDialog Ubicacion = new FolderBrowserDialog();
            if (Ubicacion.ShowDialog() == DialogResult.OK)
            {
                string Ruta = Ubicacion.SelectedPath + @"\proyectos_de_descarga.csv";
                mArchivoCsv.crearArchivoCsv(Listas.ListaProyectos, Ruta);
                MessageBox.Show("Se a creado el archivo exitosamente en "+ Ruta);
                this.Close();
            }
            else
```

```
            {
                MessageBox.Show("No se pudo crear el archivo!");
                this.Close();
            }
        }

        private void BtnCancelar_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

5.1.3.2. ARCHIVOCSV.CS

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Reflection;
using System.Text;

namespace GuardarCSV
{
    class ArchivoCsv
    {
        //metodo para crear el archivo Csv, recibe como parametro la lista y el nombre del archivo
        public void crearArchivoCsv<T>(IEnumerable<T> items, string path)
        {
            Type itemType = typeof(T);
            var props = itemType.GetProperties(BindingFlags.Public | BindingFlags.Instance).Or-
derBy(p => p.Name);
            using (var writer = new StreamWriter(path))
            {
                writer.WriteLine(string.Join(", ", props.Select(p => p.Name)));

                foreach (var item in items)
                {
                    writer.WriteLine(string.Join(", ", props.Select(p => p.GetValue(item, null))));
                }
            }
        }
    }
}
```

5.2. GENERADOR PROYECTOS DESCARGA

Diseño de Clases

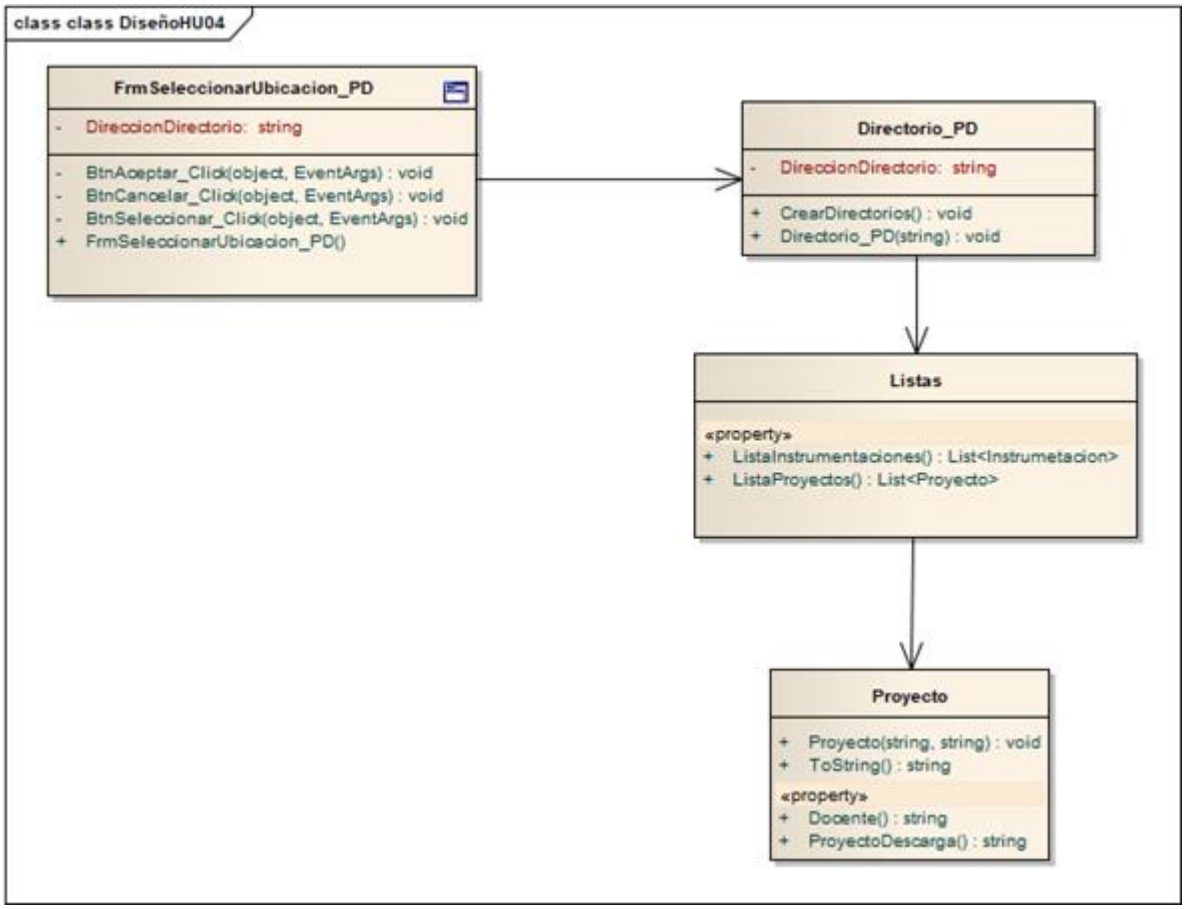
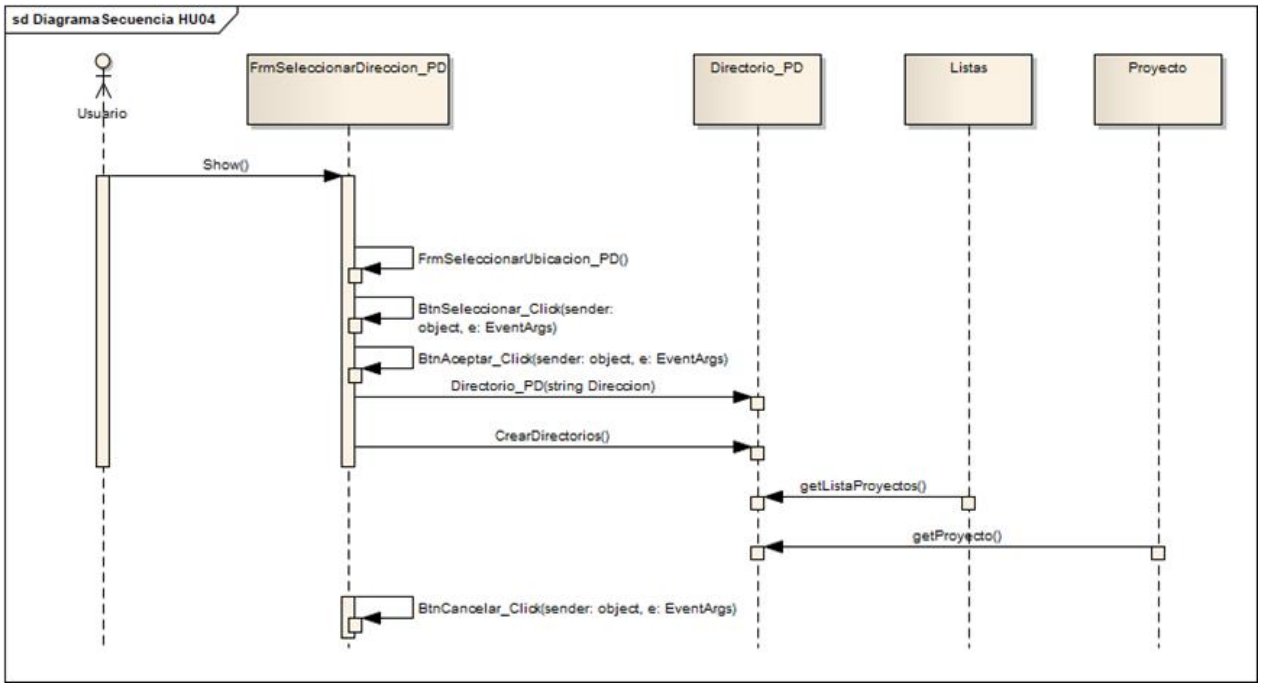
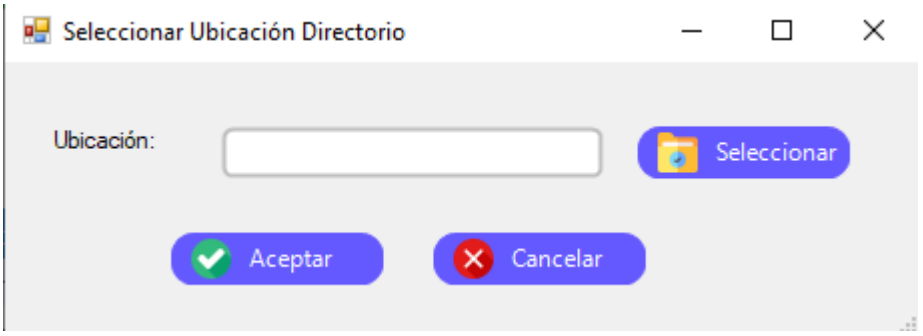


Diagrama de Secuencia



5.2.1. FRMSELECCIONARUBICACION_PD.CS




```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Collections;
using System.IO;
using SpreadsheetLight;

namespace GeneradorProyectosDesgarga
{
```

```
public partial class FrmSeleccionarUbicacion_PD : Form
{
    string DireccionDirectorio;
    public FrmSeleccionarUbicacion_PD()
    {
        InitializeComponent();
        DireccionDirectorio = "";
    }


    private void BtnSeleccionar_Click_1(object sender, EventArgs e)
    {
        // Para seleccionar la Ubicación del Directorio
        FolderBrowserDialog Ubicacion = new FolderBrowserDialog();
        if (Ubicacion.ShowDialog() == DialogResult.OK)
        {
            TxtUbicacion.Text = Ubicacion.SelectedPath;
            this.DireccionDirectorio = TxtUbicacion.Text;
        }
        else
        {
            MessageBox.Show("La ruta es Inaccesible");
        }


        if (TxtUbicacion.Text == "")
        {
            BtnAceptar.Enabled = false;
        }
        else
        {
            BtnAceptar.Enabled = true;
        }
    }

    private void BtnCancelar_Click_1(object sender, EventArgs e)
    {
        this.Close();
    }

    private void BtnAceptar_Click(object sender, EventArgs e)
    {
        //Metodo para Mandar llamar a la Clase Directorio y crearlos
        if (TxtUbicacion.Text == "")
        {
```

```
            MessageBox.Show("No se ha Seleccionado una Ubicación");
        }
        else
        {
            if (Directory.Exists(TxtUbicacion.Text))
            {
                Directorio_PD mDirectorio_PD = new Directorio_PD(this.DireccionDirectorio);

                //Ejemplo para Probar
                //this.llenarLista();

                mDirectorio_PD.CrearDirectorios();
                MessageBox.Show("El Directorio ha sido Creado");
                this.Close();
            }
            else
            {
                MessageBox.Show("{0} No es un Directorio Válido", TxtUbicacion.Text);
            }
        }
    }

    /*private void llenarLista() {
        //Para probar el Sistema
        //Inicializamos la lista en vacio
        Listas.ListaProyectos = new List<Proyecto>();

        //Creamos un registro de proyecto
        Proyecto mPro1 =
            new Proyecto("Daniel Arredondo", "Acreditación");
        //Agregamos ese registro a la lista de instrumentaciones
        Listas.ListaProyectos.Add(mPro1);

        Proyecto mPro3 =
            new Proyecto("Manuel Ignacio Salas", "Curso Arduino");
        Listas.ListaProyectos.Add(mPro3);

        //Creamos un registro de proyecto
        Proyecto mPro4 =
            new Proyecto("Daniel Arredondo", " Capacitación CDC");
        //Agregamos ese registro a la lista de instrumentaciones
        Listas.ListaProyectos.Add(mPro4);

        Proyecto mPro2 =
            new Proyecto("Manuel Ignacio Salas", "Tutorías");
```

```
            Listas.ListaProyectos.Add(mPro2);

            Proyecto mPro5 =
                new Proyecto("Daniel Arredondo", "Robótica");
            //Agregamos ese registro a la lista de instrumentaciones
            Listas.ListaProyectos.Add(mPro5);


        }*/
    }
}
```

5.2.2. DIRECTORIO_PD.CS

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;
using System.Windows.Forms;
using ListasInformacion;

namespace GeneradorProyectosDesgarga
{
    public class Directorio_PD
    {
        string DireccionDirectorio;//Ubicación de donde estará el directorio

        public Directorio_PD(string Direccion)
        {
            this.DireccionDirectorio = Direccion;
        }

        public void CrearDirectorios()
        {
            //Método para recorer la Lista e ir creando los directorios
            Listas.ListaProyectos = Listas.ListaProyectos;

            string Docente = "";
            string ProyectoDescarga = "";
            string DirExistentes = "";
            if (Listas.ListaProyectos != null)
            {
                foreach (Proyecto mProyecto in Listas.ListaProyectos)
                {
                    Docente = mProyecto.Docente;
```

```
                ProyectoDescarga = mProyecto.ProyectoDescarga;

                // Especificamos la ruta completa del directorio a crear.
                string path = @"" + this.DireccionDirectorio + "\\Proyectos_Descarga" + "\\" + Do-
cente+ "\\" + ProyectoDescarga + " ";
                try
                {
                    // Verifica si el directorio existe.
                    if (Directory.Exists(path))
                    {
                        DirExistentes+= Docente + "\\" + ProyectoDescarga + "\n";
                    }
                    else
                    {
                        //Crea la ruta completa de directorios, junto con sus subcarpetas
                        DirectoryInfo di = Directory.CreateDirectory(path);
                        Console.WriteLine("El directorio ha sido creado {0}.", Directory.GetCreation-
Time(path));
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Error: {0}", ex.ToString());
                }
            }
            if (!DirExistentes.Equals(""))
                MessageBox.Show("Los siguientes directorios ya existian: \n " + DirExistentes.Subs-
tring(0, 1000) + "...");
        }
        else
        {
            MessageBox.Show("Lista Vacia");
        }


    }

  }
}
```

6. ENTORNO DE DESARROLLO

Lenguaje de Programación utilizado:

C#

Es un lenguaje de programación multiparadigma desarrollado y estandarizado por la empresa Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA e ISO. C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común.

IDE utilizado:

Microsoft Visual Studio versión 2012 o superior.

Microsoft Visual Studio es un entorno de desarrollo integrado para Windows y macOS. Es compatible con múltiples lenguajes de programación, tales como C++, C#, Visual Basic.

KEYWORD INDEX

No se encuentran entradas de índice.