2.4 FORMULARIO PRINCIPAL

Form_Ppal.h [Diseño]

El aspecto de la interfaz de usuario de la aplicación OTOLIVE puede ser como el que vemos a continuación. Una ventana práctica y funcional, pequeña para no ocupar espacio innecesario en la pantalla, el cual será más necesario para mostrar imágenes que para ver estos menús. Con todos los botones necesarios para acceder a todas las opciones del proyecto, etc. Desde Menú se puede Abrir, crear un proyecto Nuevo, Guardar, ir a Propiedades o Salir de la aplicación completamente; desde Proyecto, se llama a las funciones principales de Información General, Calibrar Microscopio y Transecto, además de Obtener Resultados cuando se haya finalizado una lectura de un otolito (esta es la parte en la que la aplicación interpreta los datos obtenidos gracias a los conteos de los investigadores, para aportar conclusiones propias y recurrir a la estadística); por último, desde Cámara se pueden activar los modos de Video o de Captura de la cámara científica, sin necesidad de estar ejecutando otras funciones de la aplicación.

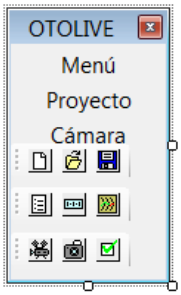

Figura 21.- Formulario Principal

Form_Ppal.h

Primero decir que los salto de líneas de código que se ven a continuación se deben a que se está obviando código que se agregó automáticamente y que no merece la pena mostrar. Por ejemplo, cuando se añade un botón, éste acarrea muchas propiedades fijadas por defecto, y normalmente ninguna o muy pocas serás usadas con algún propósito después, y si se usan, se configuran en otras partes del código que por supuesto no se ocultarán y serán debidamente explicadas.

```
1    #pragma once
2
3    // cabeceras de los dialogos secundarios
4    #include "Dlg_InfGral.h"
5    #include "Dlg_Calib.h"
6    #include "Dlg_Transect.h"
7
...  ...
14
15   // para guardar la informacion en la base de datos
16   using namespace Microsoft::Office::Interop::Excel;
17   // para acortar codigo
18   #define DlgRes System::Windows::Forms::DialogResult
19
20   namespace OTOLIVE {
21
22      /// <summary>
23      /// Resumen de Form_Ppal
24      ///
25      /// ADVERTENCIA: si cambia el nombre de esta clase, deberá cambiar la
26      ///          propiedad 'Nombre de archivos de recursos' de la herramienta de
     compilación de recursos administrados
27      ///          asociada con todos los archivos .resx de los que depende esta
     clase. De lo contrario,
28      ///          los diseñadores no podrán interactuar correctamente con los
29      ///          recursos adaptados asociados con este formulario.
30      /// </summary>
31      public ref class Form_Ppal : public System::Windows::Forms::Form
32      {
33      public:
34         // definicion de variables para comunicacion con otros procesos, internos y
     externos
35         Form_Ppal(void) :
36            // para comunicarse con la base de datos
37              exApp(gcnew Microsoft::Office::Interop::Excel::ApplicationClass())
38            // para comunicarse con InfGral
39            , dlgInfGral(gcnew Dlg_InfGral())
40            , listoInfGral(false)
41            , limpiarInfGral(false)
42            // para comunicarse con InfGral
43            , dlgCalib(gcnew Dlg_Calib())
44            , listoCalib(false)
45            , limpiarCalib(false)
46            // para comunicarse con Transect
47            //, dlgTransect(gcnew Dlg_Transect())
48            //, listoTransect(false)
49            //, limpiarTransect(false)
50         {
51            InitializeComponent();
52            //
53            //TODO: agregar código de constructor aquí
54            //
55         }
56
57      protected:
58         /// <summary>
59         /// Limpiar los recursos que se estén utilizando.
60         /// </summary>
61         ~Form_Ppal()
62         {
63            if (components)
64            {
65               delete components;
66            }
67         }
...  ...
419  #pragma endregion
420
421         // variable de comunicacion con Excel
422         Microsoft::Office::Interop::Excel::Application^ exApp;
423         // variable de comunicacion con el dialogo de informacion general
424         Dlg_InfGral^ dlgInfGral;
425         bool listoInfGral;      // true si se pulsa Aceptar
426         bool limpiarInfGral;   // true si se pulsa Limpiar
427         // variable de comunicacion con el dialogo para calibrar el microscopio
428         Dlg_Calib^ dlgCalib;
429         bool listoCalib;      // true si se pulsa Aceptar
```

```
430         bool limpiarCalib;   // true si se pulsa Limpiar
431         // para usar #define DlgRes System::Windows::Forms::DialogResult y acortar
    codigo
432         DlgRes result;
433
434         private: System::Void
    informaciónGeneralToolStripMenuItem_Click(System::Object^  sender,
    System::EventArgs^  e)
435         {
436            // abrimos el dialogo de Informacion General y guardamos lo que devuelve
437            result = dlgInfGral->ShowDialog();
438            if (result == DlgRes::OK)      // boton Aceptar -> DlgRes::OK
439               listoInfGral = true;
440            else if (result == DlgRes::Yes)   // boton Limpiar -> DlgRes::Yes
441               limpiarInfGral = true;
442         }
443
444         private: System::Void
    calibrarMicroscopioToolStripMenuItem_Click(System::Object^  sender,
    System::EventArgs^  e)
445         {
446            // abrimos el dialogo de Calibrar Microscopio y guardamos lo que devuelve
447            result = dlgCalib->ShowDialog();
448            if (result == DlgRes::OK)      // boton Aceptar -> DlgRes::OK
449               listoCalib = true;
450            else if (result == DlgRes::Yes)   // boton Limpiar -> DlgRes::Yes
451               limpiarCalib = true;
452         }
453
454         private: System::Void guardarToolStripMenuItem_Click(System::Object^  sender,
    System::EventArgs^  e)
455         {
456            // añadimos un workbook al documento de Excel creado (se crea con tres
    Worksheets)
457            Workbook^ exWb = exApp->Workbooks->Add(Type::Missing);
458            // borramos las dos ultimas worksheets
459            safe_cast<Worksheet^>(exApp->ActiveWorkbook->Sheets[3])->Delete();
460            safe_cast<Worksheet^>(exApp->ActiveWorkbook->Sheets[2])->Delete();
461            // variable para trabajar con las worksheets (por defecto empieza con la
    primera, la que tenemos)
462            Worksheet^ exWs = safe_cast<Worksheet^>(exApp->ActiveSheet);
463            // renombramos el worksheet activo (es decir, el primero)
464            exWs->Name = "Información General";
465
466            // si la Informcion General ya ha sido introducida
467            if (listoInfGral)
468            {
469               // cargamos los datos
470               exWs->Cells[ 2, 2] = "Información General";
471               exWs->Cells[ 2, 4] = dlgInfGral->NombreExp;
472               exWs->Cells[ 4, 2] = "Género";
473               exWs->Cells[ 4, 4] = dlgInfGral->Genero;
474               exWs->Cells[ 5, 2] = "Estado";
475               exWs->Cells[ 5, 4] = dlgInfGral->Estado;
476               exWs->Cells[ 6, 2] = "Corte";
477               exWs->Cells[ 6, 4] = dlgInfGral->Corte;
478               exWs->Cells[ 7, 2] = "Otolito";
479               exWs->Cells[ 7, 4] = dlgInfGral->Otolito;
480               exWs->Cells[ 8, 2] = "Especie";
481               exWs->Cells[ 8, 4] = dlgInfGral->Especie;
482               exWs->Cells[ 9, 2] = "Lectura";
483               exWs->Cells[ 9, 4] = dlgInfGral->Lectura;
484               exWs->Cells[11, 2] = "Código";
485               exWs->Cells[11, 4] = dlgInfGral->Codigo;
486               exWs->Cells[12, 2] = "Campaña";
487               exWs->Cells[12, 4] = dlgInfGral->Campana;
488               exWs->Cells[13, 2] = "Barco";
489               exWs->Cells[13, 4] = dlgInfGral->Barco;
490               exWs->Cells[14, 2] = "Número de Lance";
491               exWs->Cells[14, 4] = dlgInfGral->NumLance;
492               exWs->Cells[15, 2] = "Profundidad (m)";
493               exWs->Cells[15, 4] = dlgInfGral->Profundidad;
494               exWs->Cells[16, 2] = "Fecha del Lance";
495               exWs->Cells[16, 4] = dlgInfGral->FechaLance;
496               exWs->Cells[17, 2] = "Tipo de Peso";
497               exWs->Cells[17, 4] = dlgInfGral->RdPeso;
498               exWs->Cells[18, 2] = "Peso (gr)";
```

```
499                exWs->Cells[18, 4] = dlgInfGral->Peso;
500                exWs->Cells[19, 2] = "Longitud Total (mm)";
501                exWs->Cells[19, 4] = dlgInfGral->Longitud;
502                exWs->Cells[21, 2] = "Fecha de la Lectura";
503                exWs->Cells[21, 4] = dlgInfGral->FechaLec;
504                exWs->Cells[22, 2] = "Número de Lecturas";
505                exWs->Cells[22, 4] = dlgInfGral->NumLec;
506                exWs->Cells[23, 2] = "Investigador";
507                exWs->Cells[23, 4] = dlgInfGral->NombreInv;
508                exWs->Cells[25, 2] = "Observaciones";
509                exWs->Cells[25, 4] = dlgInfGral->Observaciones;
510             }
511             else if (limpiarInfGral)   // si se quiere limpiar
512             {
513                // limpiamos los datos
514                exWs->Cells[ 2, 2] = "Información General";
515                exWs->Cells[ 2, 4] = "";
516                exWs->Cells[ 4, 2] = "Género";
517                exWs->Cells[ 4, 4] = "";
518                exWs->Cells[ 5, 2] = "Estado";
519                exWs->Cells[ 5, 4] = "";
520                exWs->Cells[ 6, 2] = "Corte";
521                exWs->Cells[ 6, 4] = "";
522                exWs->Cells[ 7, 2] = "Otolito";
523                exWs->Cells[ 7, 4] = "";
524                exWs->Cells[ 8, 2] = "Especie";
525                exWs->Cells[ 8, 4] = "";
526                exWs->Cells[ 9, 2] = "Lectura";
527                exWs->Cells[ 9, 4] = "";
528                exWs->Cells[11, 2] = "Código";
529                exWs->Cells[11, 4] = "";
530                exWs->Cells[12, 2] = "Campaña";
531                exWs->Cells[12, 4] = "";
532                exWs->Cells[13, 2] = "Barco";
533                exWs->Cells[13, 4] = "";
534                exWs->Cells[14, 2] = "N° de Lance";
535                exWs->Cells[14, 4] = "";
536                exWs->Cells[15, 2] = "Profundidad (m)";
537                exWs->Cells[15, 4] = "";
538                exWs->Cells[16, 2] = "Fecha del Lance";
539                exWs->Cells[16, 4] = "";
540                exWs->Cells[17, 2] = "Tipo de Peso";
541                exWs->Cells[17, 4] = "";
542                exWs->Cells[18, 2] = "Peso (gr)";
543                exWs->Cells[18, 4] = "";
544                exWs->Cells[19, 2] = "Longitud Total (mm)";
545                exWs->Cells[19, 4] = "";
546                exWs->Cells[21, 2] = "Fecha de la Lectura";
547                exWs->Cells[21, 4] = "";
548                exWs->Cells[22, 2] = "Número de Lecturas";
549                exWs->Cells[22, 4] = "";
550                exWs->Cells[23, 2] = "Investigador";
551                exWs->Cells[23, 4] = "";
552                exWs->Cells[25, 2] = "Observaciones";
553                exWs->Cells[25, 4] = "";
554                limpiarInfGral = false;
555             }
556
557             // si ya se ha realizado la calibracion
558             if (listoCalib)
559             {
560                // cargamos los datos
561                exWs->Cells[ 2, 6] = "Microscopio";
562                exWs->Cells[ 4, 6] = "Ocular";
563                exWs->Cells[ 4, 8] = "x" + dlgCalib->Ocular;
564                exWs->Cells[ 5, 6] = "Objetivo";
565                exWs->Cells[ 5, 8] = "x" + dlgCalib->Objetivo;
566                exWs->Cells[ 6, 6] = "Aumentos";
567                exWs->Cells[ 6, 8] = "x" + dlgCalib->Aumentos;
568                exWs->Cells[ 7, 6] = "Píxeles medidos";
569                exWs->Cells[ 7, 8] = dlgCalib->Pixeles;
570                exWs->Cells[ 8, 6] = "Unidades";
571                exWs->Cells[ 8, 8] = dlgCalib->Unidades;
572                if (dlgCalib->Unidades == "Milímetros")
573                   exWs->Cells[ 9, 6] = "Micrómetro (mm)";
574                else
575                   exWs->Cells[ 9, 6] = "Micrómetro (um)";
```

```
576             exWs->Cells[ 9, 8] = dlgCalib->Micrometro;
577             if (dlgCalib->Unidades == "Milímetros")
578                exWs->Cells[10, 6] = "Relación (mm/pixel)";
579             else
580                exWs->Cells[10, 6] = "Relación (um/pixel)";
581             exWs->Cells[10, 8] = dlgCalib->Relacion;
582             exWs->Cells[11, 6] = "Observaciones";
583             if (dlgCalib->Observaciones == "Observaciones (opcional)")
584                exWs->Cells[11, 8] = "";
585             else
586                exWs->Cells[11, 8] = dlgCalib->Observaciones;
587          }
588          else if (limpiarCalib)   // si se quiere limpiar
589          {
590             exWs->Cells[ 2, 6] = "Microscopio";
591             exWs->Cells[ 4, 6] = "Ocular";
592             exWs->Cells[ 4, 8] = "";
593             exWs->Cells[ 5, 6] = "Objetivo";
594             exWs->Cells[ 5, 8] = "";
595             exWs->Cells[ 6, 6] = "Aumentos";
596             exWs->Cells[ 6, 8] = "";
597             exWs->Cells[ 7, 6] = "Unidades";
598             exWs->Cells[ 7, 8] = "";
599             exWs->Cells[ 8, 6] = "Micrómetro";
600             exWs->Cells[ 8, 8] = "";
601             exWs->Cells[ 9, 6] = "Píxeles medidos";
602             exWs->Cells[ 9, 8] = "";
603             exWs->Cells[10, 6] = "Relación (uds/pixel)";
604             exWs->Cells[10, 8] = "";
605             exWs->Cells[11, 6] = "Observaciones";
606          }
607          // mostramos el documento de Excel
608          exApp->Visible = true;
609       }
610
611       private: System::Void salirToolStripMenuItem_Click(System::Object^  sender,
    System::EventArgs^  e)
612       {
613          // para salir
614          System::Windows::Forms::Application::Exit();
615       }
616
617       private: System::Void videoToolStripMenuItem_Click(System::Object^  sender,
    System::EventArgs^  e)
618       {
619          // para activar el video
620       }
621
622       private: System::Void capturarImagenToolStripMenuItem_Click(System::Object^
    sender, System::EventArgs^  e)
623       {
624          // para capturar una imagen
625       }
626 };
627 }
```

Esencialmente, este Formulario Principal no es más que el telón de fondo de toda la aplicación OTOLIVE, ya que la consola permanece oculta y le ha concedido todo el control del programa, ya que simplemente lo inicia, después es el formulario el que maneja todas las funciones, como se muestra en la siguiente estructura de archivos.

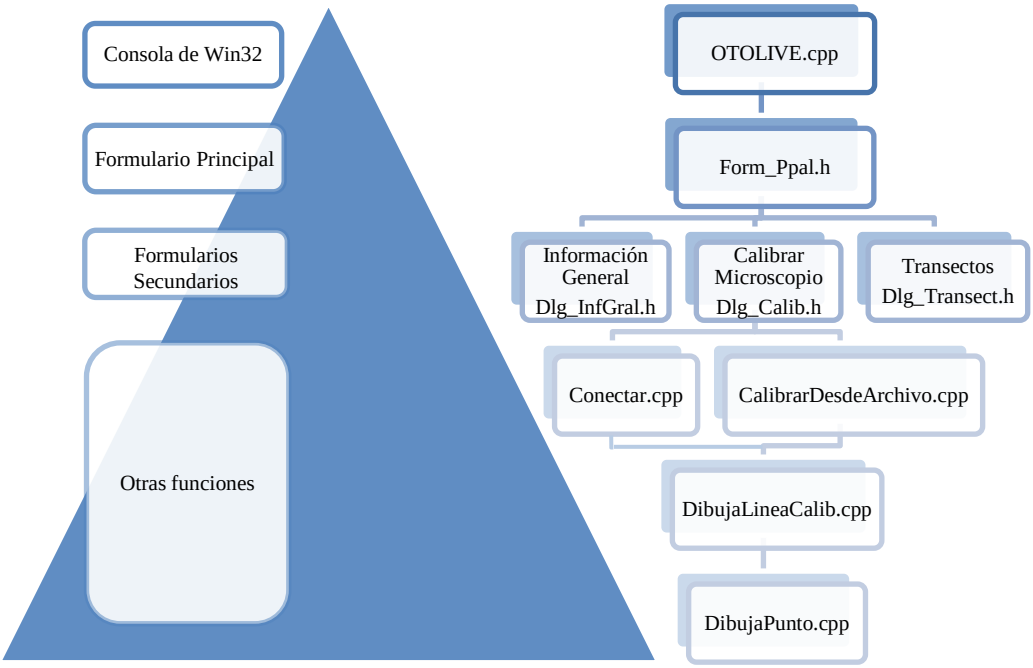

Figura 22.- Orden jerárquico de los archivos del programa

Por eso, lo primero que nos encontramos en las líneas 4 y 5, son las cabeceras de los formularios secundarios, a los cuales llama gracias las líneas 39 y 43 dentro del constructor de Form_Ppal.h. Ya sólo queda declarar sus respectivas variables en las líneas 424 y 428 para poder ejecutarlos desde las líneas 437 y 447 respectivamente para Información General y para Calibrar Microscopio, por medio del comando ShowDialog() dentro de las funciones de evento de los botones que conectan con cada uno de ellos. Esta es la manera de que un formulario pueda invocar a otros.

Para estructurar el código de un formulario cualquiera atendiendo a las partes que nos interesan, podemos empezar por la "cabecera", donde ponemos todos los `#include`, los `using namespace` y los `#define`. Después vendría el "constructor", donde se inicializan las variables del formulario. Ahora nos encontramos con una gran cantidad de código que ha sido generado automáticamente al ir creando la interfaz del formulario, por lo que damos un gran salto hasta `#pragma endregion`, a partir de donde empezamos a escribir nuestro código, que es la última y más importante parte de nuestro programa.

Este código estará compuesto fundamentalmente por gran cantidad de pequeñas funciones, que son ni más ni menos que los manejadores de eventos de los componentes

de nuestro formulario (botones, cuadros de texto, botones de opción, listas desplegables, etc.). Lo que más a menudo nos vamos a encontrar, son eventos tipo "qué hacer cuando se hace clic sobre un botón", pero existen muchas más opciones y las podemos ver en la Ventana de Propiedades, en la pestaña de Eventos. Los tres eventos que hay programados son los de los botones para abrir el formulario de Información General, el de Calibrar Microscopio, y el de Guardar todo en la base de datos.

Las acciones de los botones para llamar a los formularios secundarios son iguales en estructura, ya que primero los abre, pasando a tener éstos el control de la aplicación mientras realizan sus funciones, y cuando se cierran (ya sea por el botón Aceptar, Limpiar o Cancelar), el control vuelve al Formulario Principal, y éste recibe un valor devuelto por los otros formularios al cerrarse, un resultado que nos servirá para actuar de una u otra forma. Por eso si el valor devuelto es el del botón Aceptar, significa que todos los campos del formulario secundario han sido introducidos correctamente, y se activa un flag para poder guardar esos datos en la base de datos. Si el valor devuelto proviene del botón Limpiar, se hace lo contrario, se activa un flag que borra la base de datos de ese formulario concreto. Y por último, el botón cancelar no genera ninguna acción sobre la base de datos, como es lógico. La siguiente figura muestra este concepto.

Figura 23.- Acción de Guardar/Limpiar la base de datos

La función que queda es la más importante, la del evento de pulsar el botón Guardar. Como hemos dicho, este evento reacciona de manera distinta si el valor devuelto por los formularios secundarios es el de Aceptar o el de Limpiar. Pues bien, el primero rellena los campos de la base de datos con lo que se ha introducido en los formularios, y el segundo los borra. Ya sólo queda ver cómo nos comunicamos con la base de datos.

Ya se agregó al crear el proyecto la referencia Microsoft.Office.Interop.Excel 11.0.0.0[8], y ahora en nuestro formulario principal, hay que añadir la línea 16 en las cabeceras, la 37 en el constructor, y la 422 para definir la variable de comunicación con la base de datos. Y gracias a los conceptos aprendidos en un tutorial se puede incluir el código para preparar el documento Excel de las líneas 457 a 464. Después, lo que nos encontramos dentro de las condiciones manejadas por los flags, es el volcado de datos en estado puro, tanto almacenamiento como borrado, accediendo al contenido de las

[8]Tutorial para comunicación Excel-Visual Studio

celdas de Excel. Y finalmente en la línea 608 es donde se muestra el documento, para ser guardado posteriormente.

Antes de pasar a explicar el siguiente archivo del proyecto, hay que hacer un apunte importante, y es para hablar sobre la función IntelliSense que posee Visual Studio. Es una de las mayores ventajas para la programación, ya que al escribir nuestro código, IntelliSense nos muestra todas las posibles subclases y miembros que puede tener una variable, con todos sus parámetros de entrada y salida, etc. Por eso cuando no está disponible esta función y además en difícil encontrar información relevante en la red, se frena mucho la marcha de desarrollo en la programación.

Y se señala lo anterior porque algo como tan simple usar un `#define Microsoft::Office::Interop::Excel::Application Excel` para acortar código y poder escribir `Excel^ exApp` en lugar de lo que hay en la línea 422, provocó el que no se pudiera disponer de la ayuda de IntelliSense para las funciones de Excel durante gran parte del desarrollo del proyecto, por lo que no se pudo avanzar demasiado en conceptos necesarios como abrir documentos ya existentes para tomar datos, o seguir creando más hojas para ir almacenando la información de cada transecto, etc. Entonces, hasta que se cae en la cuenta de que algo que no debería dar ningún fallo está perjudicando al proyecto, se pierde un tiempo precioso. Queda entonces para otros desarrolladores la labor de seguir avanzando en esta rama de la comunicación con la base de datos.