# MANUAL DE DESARROLLO DE LOADERS ELLIPSE 8.4

# HÉCTOR J. HERNÁNDEZ R.

# Ingeniero de Sistemas

# Hambings S.A.S

# Versión 1.3

# 15 de Marzo de 2019

TABLA DE CONTENIDO

[MANUAL DE DESARROLLO DE LOADERS ELLIPSE 8.4 1](#_Toc3244022)

[HÉCTOR J. HERNÁNDEZ R. 1](#_Toc3244023)

[Ingeniero de Sistemas 1](#_Toc3244024)

[Hambings S.A.S 1](#_Toc3244025)

[Versión 1.3 1](#_Toc3244026)

[15 de Marzo de 2019 1](#_Toc3244027)

[PALABRAS CLAVES 3](#_Toc3244028)

[REQUISITOS PREVIOS: 3](#_Toc3244029)

[PROCEDIMIENTO 3](#_Toc3244030)

[1. CREAR NUEVO PROYECTO 3](#_Toc3244031)

[2. CONSTRUCCIÓN DE LA ESTRUCTURA GENERAL DEL PROYECTO 5](#_Toc3244032)

[2.1 Crear Cinta de Opciones (Ribbon) 5](#_Toc3244033)

[2.2 Renombrar Elementos de la Cinta de Opciones (Ribbon) 7](#_Toc3244034)

[2.3 Adicionar controles a la Cinta de Opciones (Ribbon) 8](#_Toc3244035)

[3. AGREGAR REFERENCIA A ORACLE DATA PROVIDER 11](#_Toc3244036)

[3.1 Instalación 11](#_Toc3244037)

[3.1.1 Instalar por NuGet del ODP.Net 11](#_Toc3244038)

[3.1.2 Instalación Manual 13](#_Toc3244039)

[Agregar referencia a la librería 13](#_Toc3244040)

[Agregar archivo de configuración app.config 15](#_Toc3244041)

[3.2 Agregar referencias TNS/DataSources de acceso a bases de datos 17](#_Toc3244042)

[3.3 Agregar código de referencia de Oracle Data Manager 19](#_Toc3244043)

[4. AGREGAR REFERENCIA A LIBRERÍAS COMUNES 20](#_Toc3244044)

[4.1 Vincular el proyecto EllipseCommonsClassLibrary 20](#_Toc3244045)

[4.2 Agregar código de referencias de Librerías Commons 22](#_Toc3244046)

[5. AGREGAR CABECERA DE CONSUMO SOAP 24](#_Toc3244047)

[5.1 Agregar referencia a librería de consumo SOAP 24](#_Toc3244048)

[5.2 Agregar código de referencia de encabezado SOAP en el proyecto 25](#_Toc3244049)

[5.3 Adicionar referencia a app.config 25](#_Toc3244050)

[6. AGREGAR REFERENCIAS WEB Y CUALQUIER OTRA REFERENCIA REQUERIDA 26](#_Toc3244051)

[6.1 Agregar código de referencia del Screen Service 26](#_Toc3244052)

[6.2 Agregar Web Service de Ellipse 26](#_Toc3244053)

[6.3 Agregar referencia de Web Service de Ellipse en el Proyecto 27](#_Toc3244054)

[7. OBSERVACIONES ADICIONALES DE DESARROLLO 27](#_Toc3244055)

[7.1 Autenticación 27](#_Toc3244056)

[7.2 Uso del servicio seleccionado 27](#_Toc3244057)

[7.3 Ejecución de Querys 28](#_Toc3244058)

[7.4 Control de Errores y Debugging 29](#_Toc3244059)

[7.5 Validaciones de Campo 30](#_Toc3244060)

[7.6 Gestión de múltiples hojas de Excel 32](#_Toc3244061)

[7.7 Manejo de hilos de procesos 33](#_Toc3244062)

[8. CONTROL DE VERSIONES Y PUBLICACIÓN DEL PROYECTO 35](#_Toc3244063)

[8.1 Control de versiones 35](#_Toc3244064)

# PALABRAS CLAVES

Ellipse, VSTO, Office, Excel, AddIn, WebService, Loader, Macro, SOAP

# REQUISITOS PREVIOS:

1. Versión de Visual Studio compatible con VSTO para Office 2010
2. Microsoft Office 2010 o superior
3. .Net FrameWork 4.0
4. Acceo a la Red Interna

# PROCEDIMIENTO

## CREAR NUEVO PROYECTO

Desplegamos el menú principal y seleccionamos “[1] FILE > [2] New > [3] Project …”

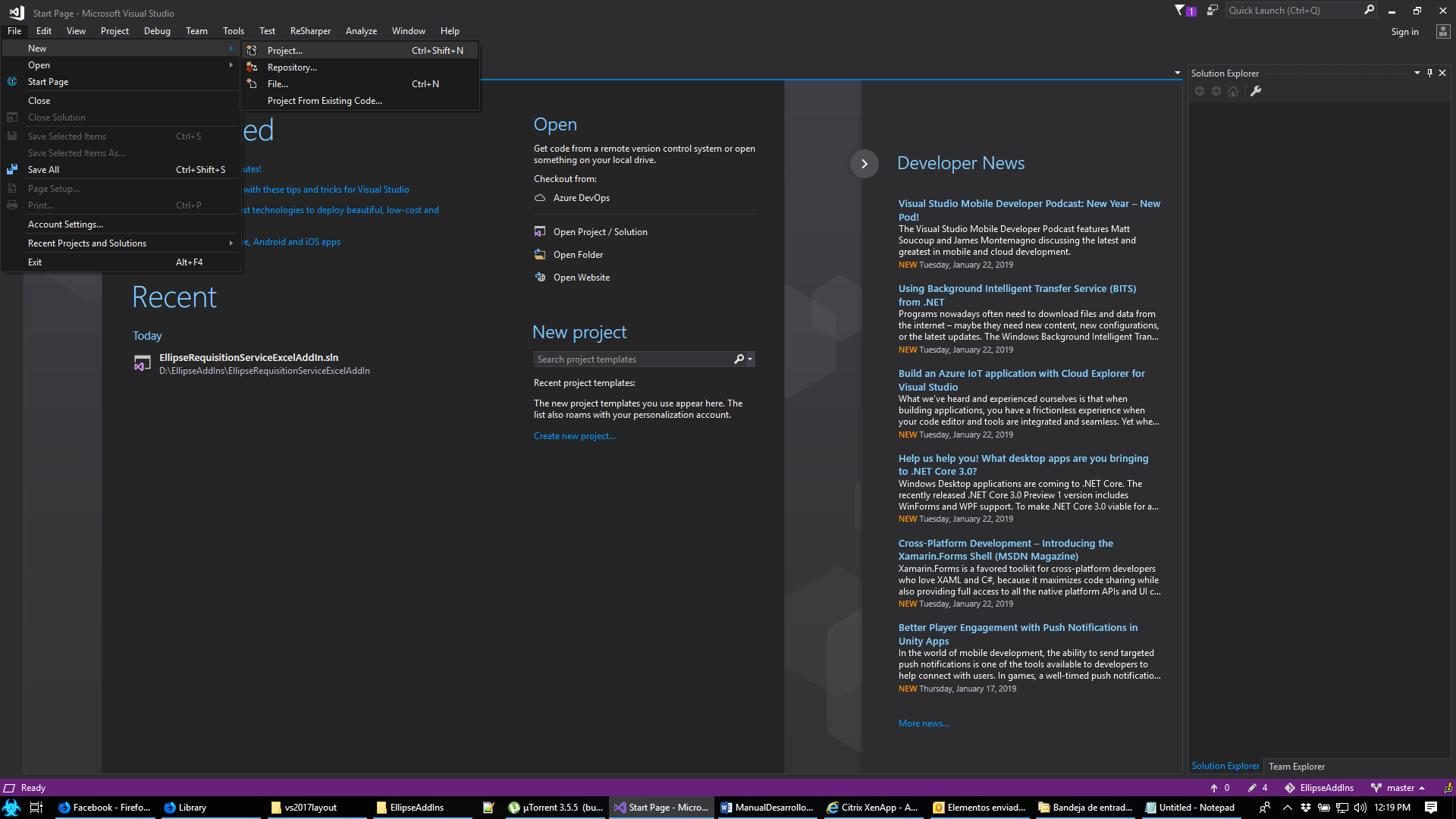


Imagen . Nuevo Proyecto: Creación de Nuevo proyecto

Seleccionamos la plantilla correspondiente a “Excel 2010 AddIn” en las opciones laterales, “[4] Visual C# > [5] Office/SharePoint > [6] VSTO Add-Ins > [7] Excel 2010 AddIn”.

[8] Nos aseguramos que la versión del .NET Framework seleccionada sea la versión 4 para garantizar la compatibilidad con el Excel 2010 utilizado en Cerrejón.

[9] Ingresamos el Nombre del proyecto como **EllipseProyectoExcelAddIn**, respetando la convención establecida para nombres de desarrollo.

[10] Establecemos las opciones del repositorio y de la jerarquía de directorios según se requiera y presionamos el botón de OK.

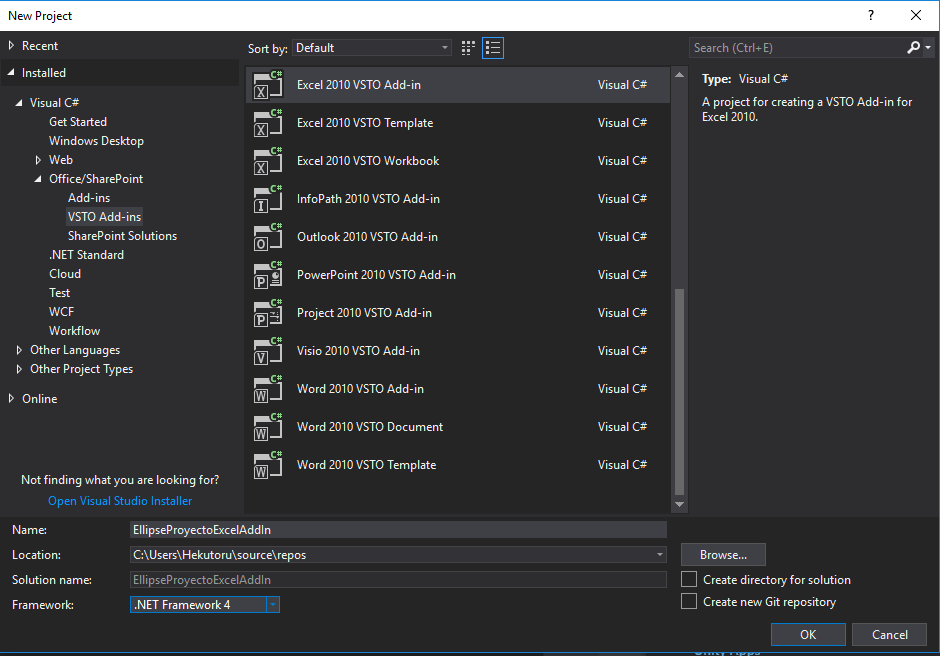


Imagen . Nuevo Proyecto: Selección de Tipo de Proyecto

## CONSTRUCCIÓN DE LA ESTRUCTURA GENERAL DEL PROYECTO

### Crear Cinta de Opciones (Ribbon)

Desplegamos el [1] menú de opciones del proyecto sea desde el menú principal o haciendo clic derecho en el explorador de soluciones, y seleccionamos “[2] Add > [3] New Item…”

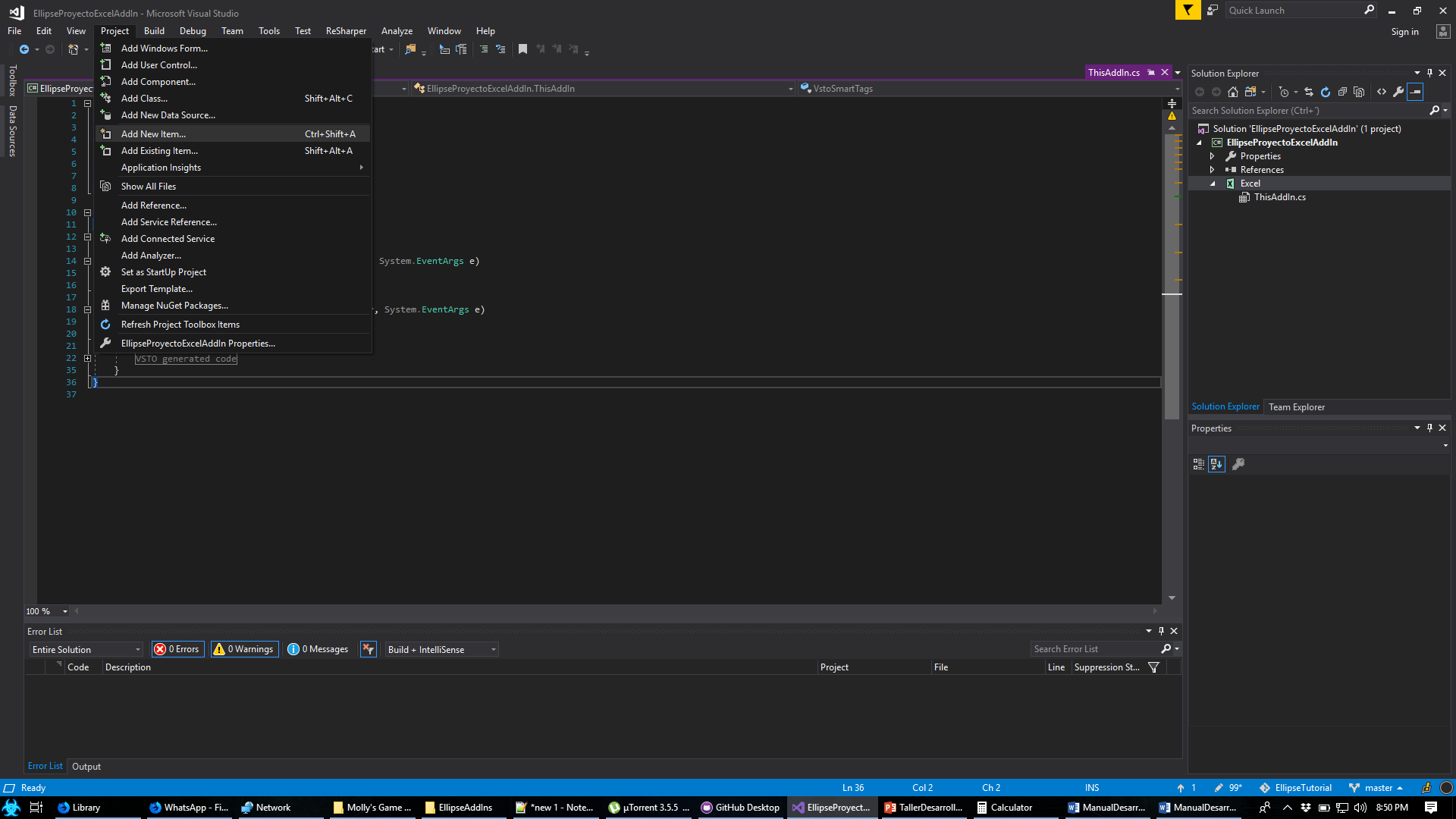
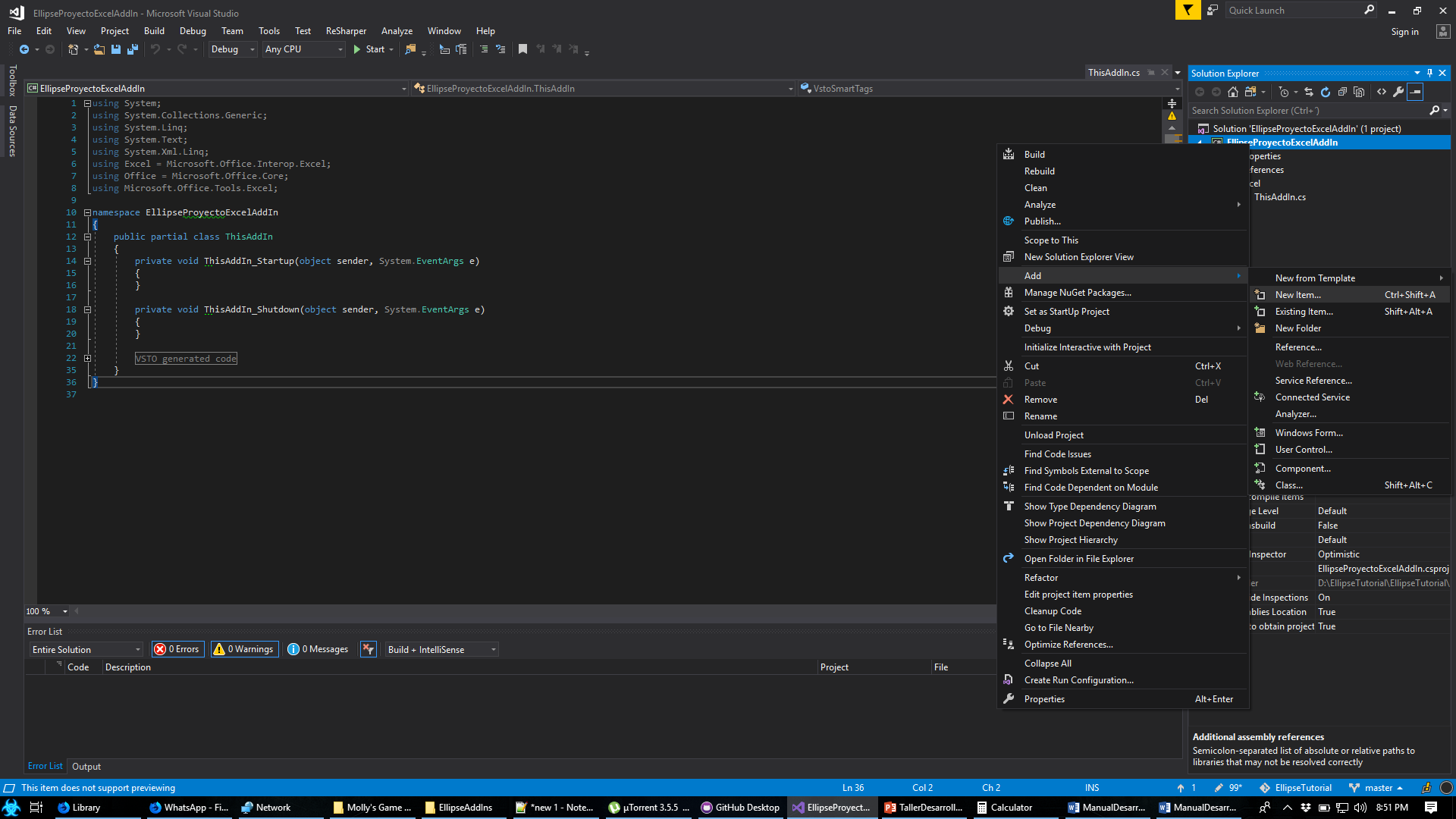


Imagen . Crear Cinta de Opciones (Ribbon): Nuevo Ítem

En el cuadro desplegado seleccionamos el elemento de la categoría “[4] Visual C# Items” y “[5] Office/SharePoint” el elemento “[6] Ribbon (Visual Designer)”. Le damos nombre al Ribbon a adicionar [7] según la convención establecida “RibbonEllipse.cs” y le damos al “[8] botón Add”

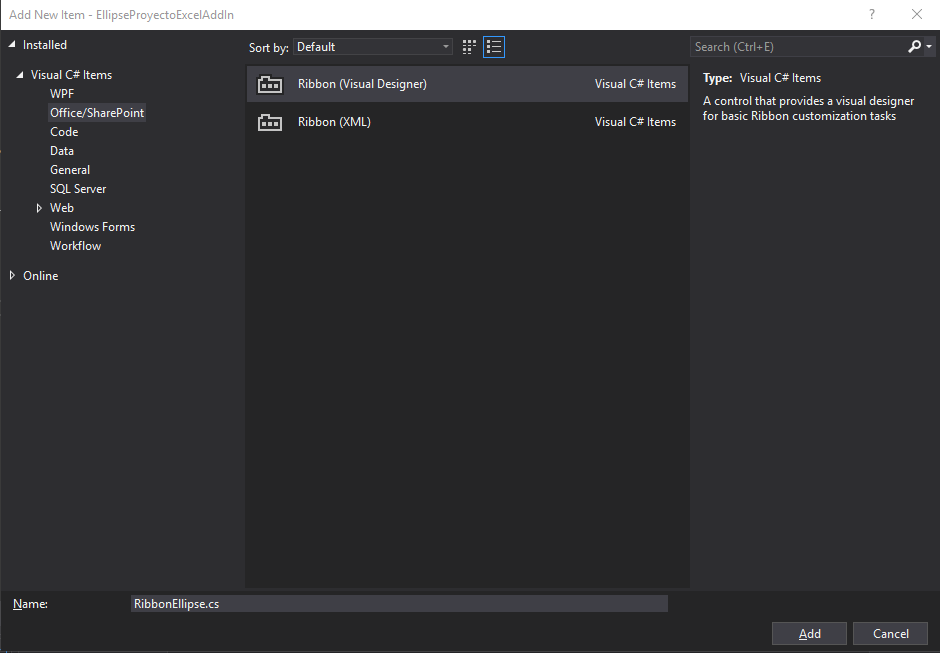


Imagen . Crear Cinta de Opciones (Ribbon): Seleccionar tipo de Item

### Renombrar Elementos de la Cinta de Opciones (Ribbon)

Seleccionamos la pestaña del Ribbon [1] y cambiamos el texto de la etiqueta [2] a “ELLIPSE 8” y también cambiamos el nombre [3] con el que vamos a hacer referencia en nuestro código a “tabEllipse. Esta acción permitirá agrupar todos los AddIns que se desarrollen, bajo la misma pestaña de opciones.

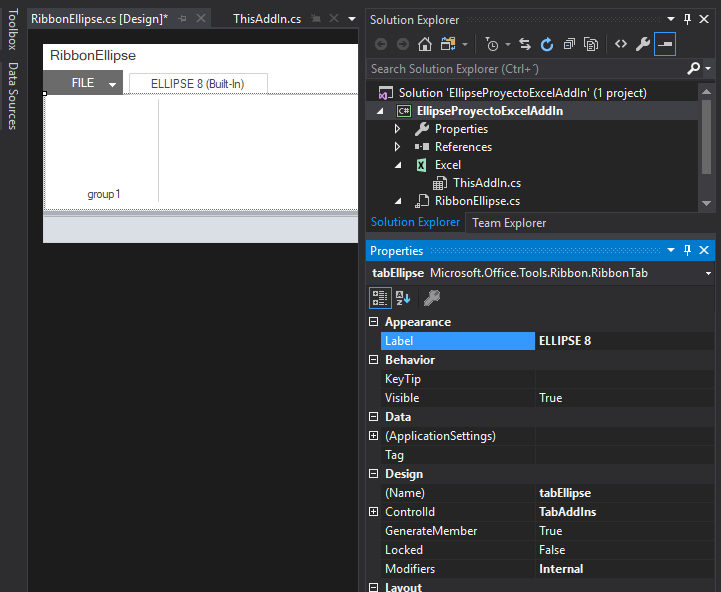


Imagen . Renombrar Cinta de Opciones (Ribbon): Renombrar Pestaña

Repetimos el mismo proceso para el grupo del Ribbon. Seleccionamos el grupo del Ribbon [4] y cambiamos el texto de la etiqueta al nombre del proyecto [5] “Proyecto” y también cambiamos el nombre [6] con el que vamos a hacer referencia en nuestro código a “grpProyecto”.

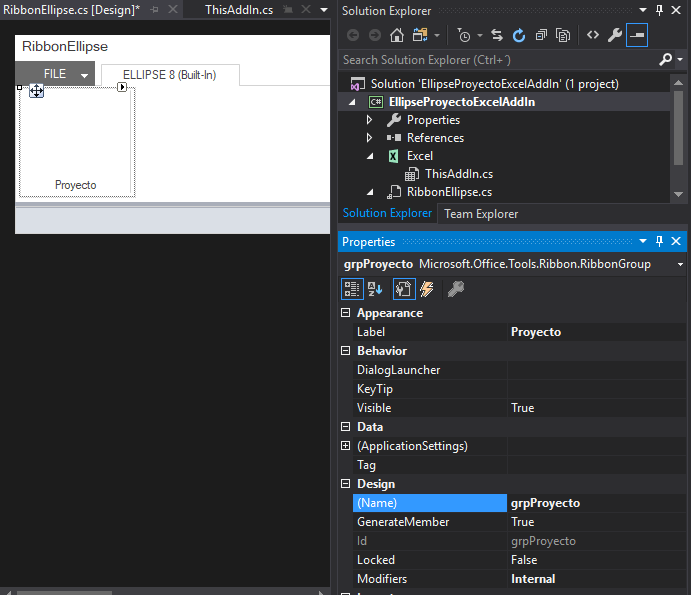


Imagen . Renombrar Elementos de la Cinta de Opciones (Ribbon)

### Adicionar controles a la Cinta de Opciones (Ribbon)

Desde la barra lateral izquierda, desplegamos las opciones de la “[1] Caja de Herramientas (Toolbox)” y del menú “[2] Office Ribbon Controls” añadiremos los siguientes elementos según como se ve en la imagen:

* Box
* Button (Dentro del elemento Box “button1”)
* Button (Dentro del elemento Box “button2”)
* DropDown
* Button (“button3”)

Nota: Según la cantidad de opciones y/o acciones que se requiera, el botón 3 deberá ser reemplazado por un menú de opciones, y dentro de este se adicionarán los botones respectivos para cada opción/acción.

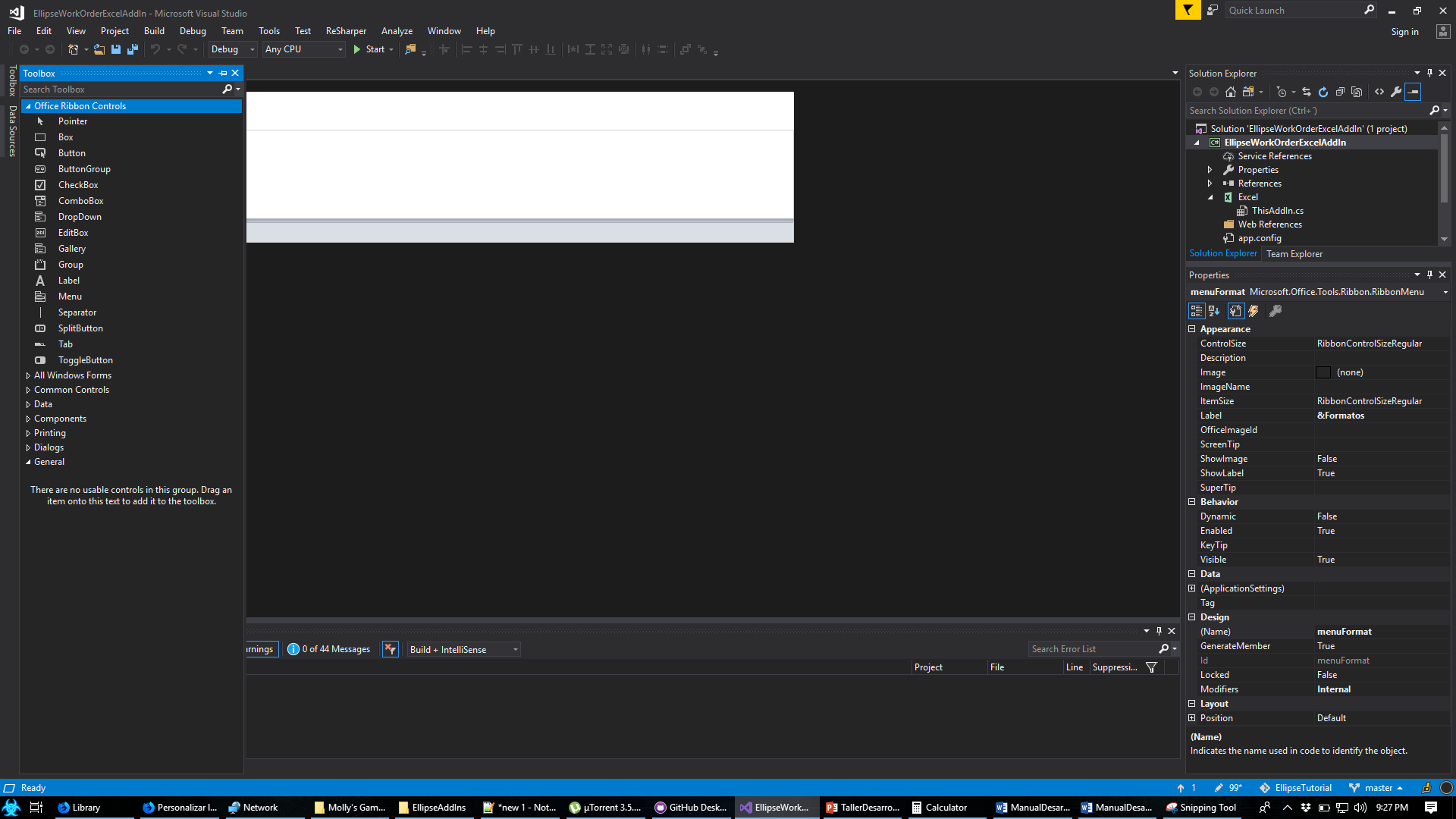


Imagen . Adicionar controles a la Cinta de Opciones (Ribbon): Selección de Controles

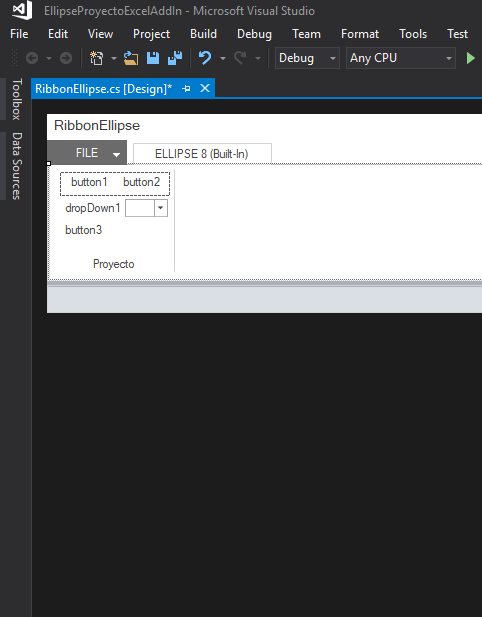


Imagen . Adicionar controles a la Cinta de Opciones (Ribbon): Controles Añadidos

Seleccionamos cada uno de los elementos y modificamos los valores en la barra de propiedades, según como lo vemos en la siguiente tabla.

|  |  |  |
| --- | --- | --- |
| Elemento  (Item) | Etiqueta  (Label) | Nombre  (Name) |
| button1 | Formatear | btnFormat |
| button2 | ? | btnAbout |
| dropDown1 | Env. | drpEnvironment |
| button3 | Ejecutar | btnExecution |

Tabla : Renombrar valores de Controles de Cinta de Opciones (Ribbon)

Nota: Para asignar un atajo de tecla a algún elemento, es suficiente con añadir el símbolo ampersand “&” antes de la letra del atajo en los valores de Label.

El resultado esperado debe verse como lo muestra la siguiente imagen.

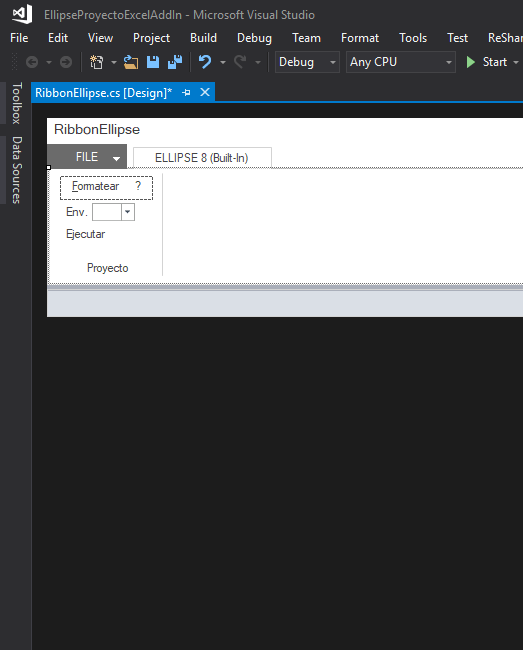


Imagen . Renombrar Controles de la Cinta de Opciones (Ribbon)

Nota: La lista de los servidores para la lista desplegable de Entorno se agregará de forma automática en la función de carga del Ribbon.

## AGREGAR REFERENCIA A ORACLE DATA PROVIDER

La librería de Oracle Data Provider corresponde a la librería para acceder a las bases de datos tipo Oracle. Esta librería puede ser añadida directamente desde el repositorio NuGet para instalar la última versión o puede añadirse de forma manual la versión deseada.

### Instalación

### Instalar por NuGet del ODP.Net

Nota: Si va a hacer uso de la librería EllipseCommonsClassLibrary se recomienda omitir este paso y seguir con la instalación manual.

Para instalar la librería Oracle ODP.NET Managed Drivers nos dirigimos al menú de “[1] PROJECT > [2] Manage NuGet Packages…”

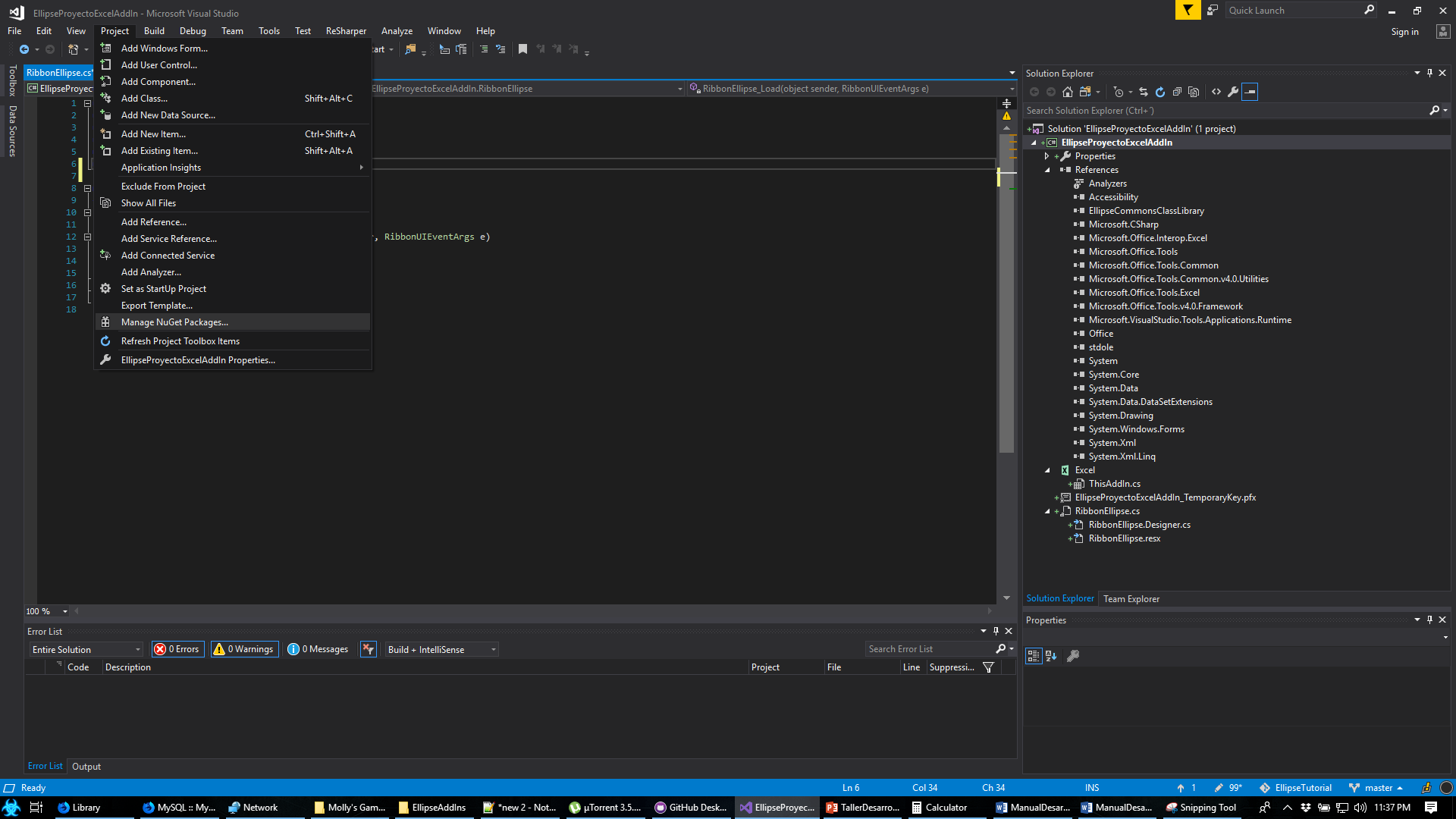


Imagen . Agregar Oracle Data Provider

Buscamos [3] Online, y escribimos como criterio [4] *“Oracle ODP”*. Seleccionamos el resultado como se muestra en la imagen y le damos al [5] botón Instalar.

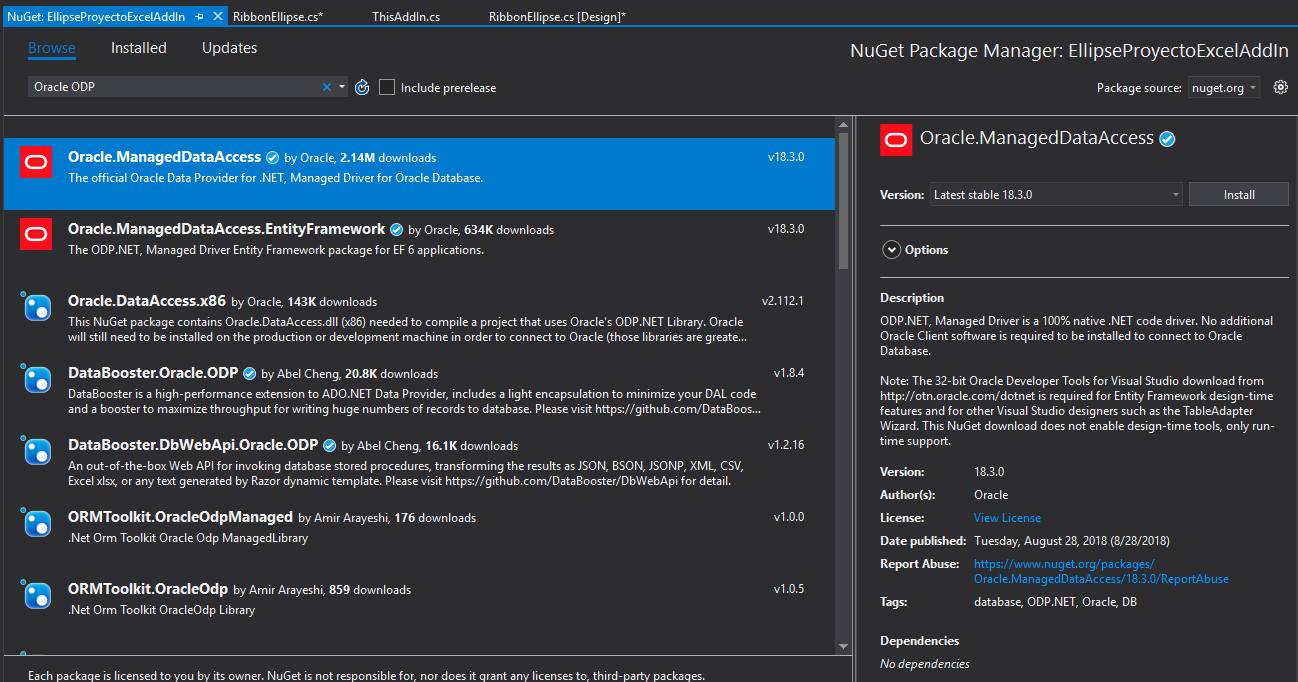


Imagen . Agregar Oracle Data Provider: Instalar la referencia

Finalmente, seguimos el proceso de aceptación de licencia y comprobamos al final que esté instalada la librería corroborando que el ícono verde de verificación esté sobre la librería instalada

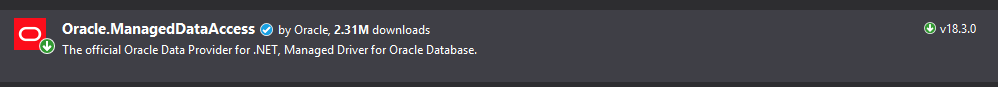


Imagen . Agregar Oracle Data Provider: Verificar el estado

### Instalación Manual

Nota: Si realizó la instalación por el paquete NuGet, puede omitir este paso.

### Agregar referencia a la librería

Desplegamos el [1] menú de opciones del proyecto sea desde el menú principal o haciendo clic derecho en el explorador de soluciones, y seleccionamos “[2] Agregar (Add) > [3] Referencia (Reference)”

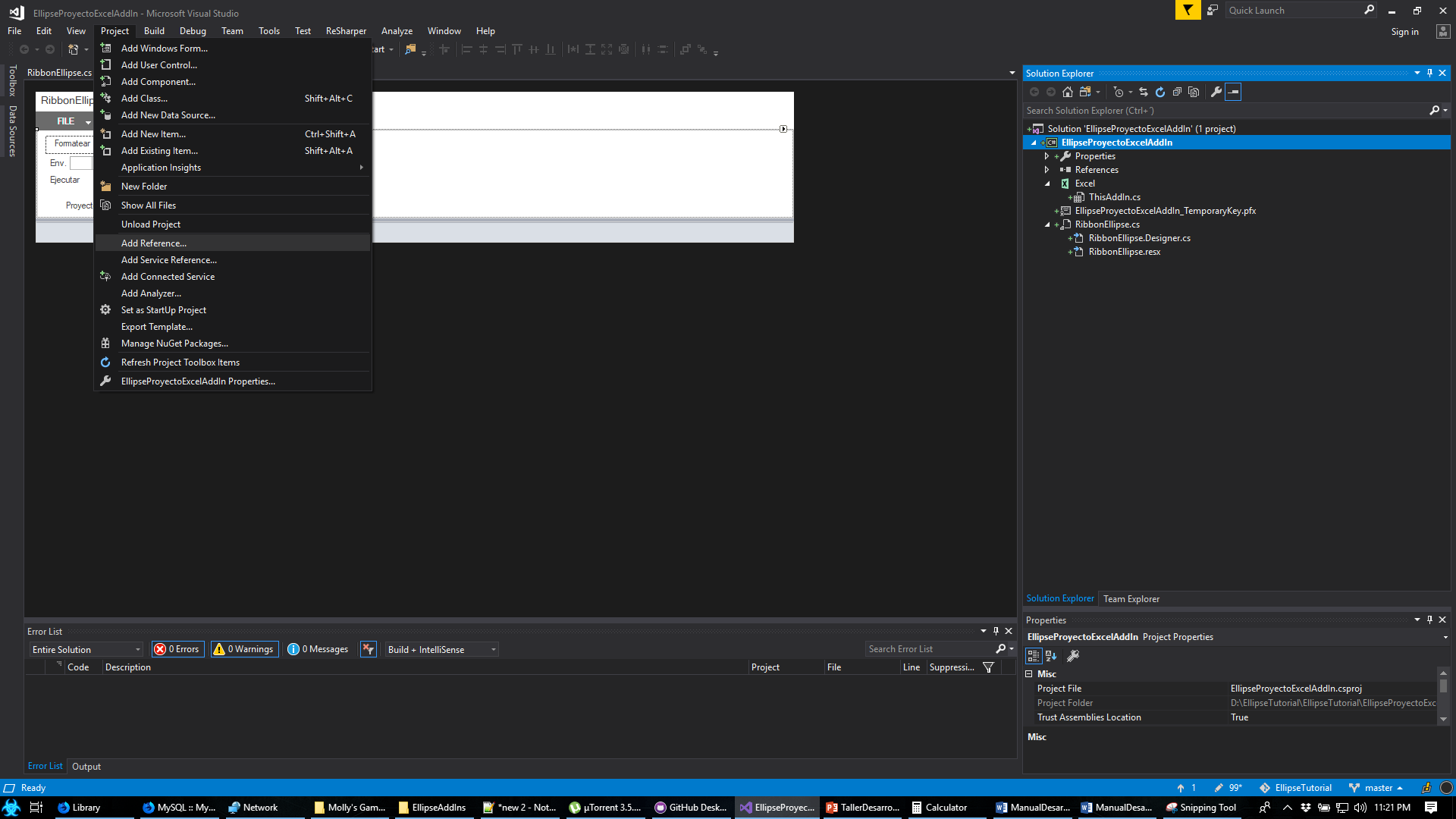
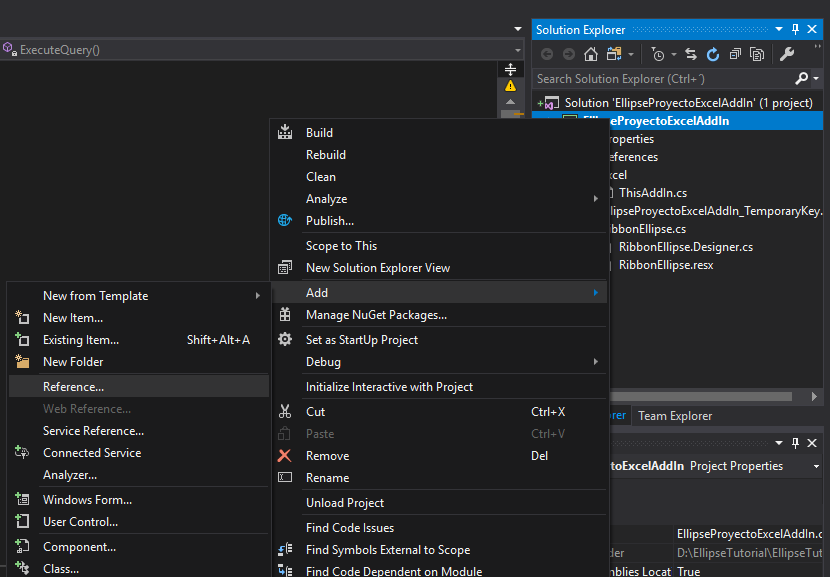


Imagen . Adicionar librería ODP: Nueva referencia

En la barra lateral izquierda seleccionamos la opción de [4] Búsqueda (Browse) y nuevamente en el botón de [5] Búsqueda (Browse). Ubicamos la ruta donde se encuentra el archivo de librería con la versión correspondiente, seleccionamos [6] Oracle.ManagedDataAccess.dll y hacemos clic en el botón de [7] Añadir (Add).

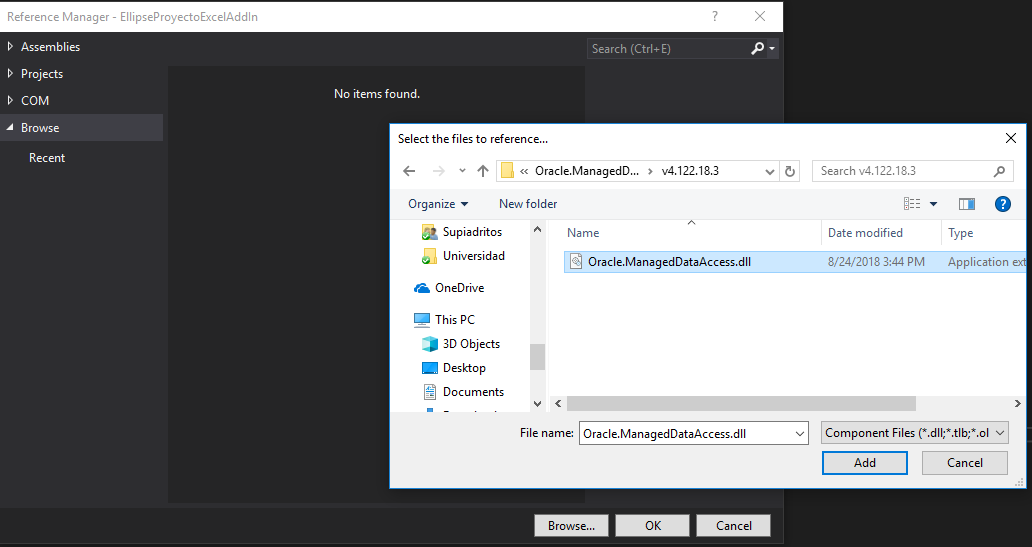


Imagen . Adicionar librería ODP: Seleccionar ruta

Nos aseguramos que el cuadro de selección [8] esté activo en el listado y finalizamos haciendo clic en el botón de [9] OK

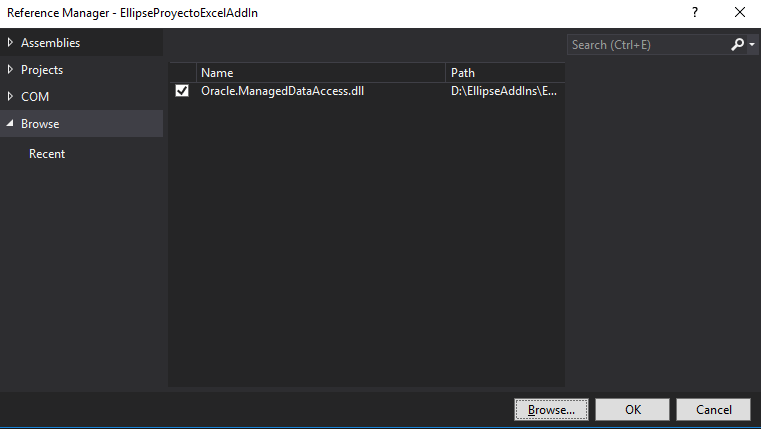


Imagen . Adicionar librería ODP: Seleccionar librería

### Agregar archivo de configuración app.config

Desplegamos el menú de [1] Proyecto (Project) en el menú principal o haciendo clic derecho en nuestro proyecto, seleccionamos la opción de [2] Agregar (Add) y [3] Nuevo Ítem (New Item)

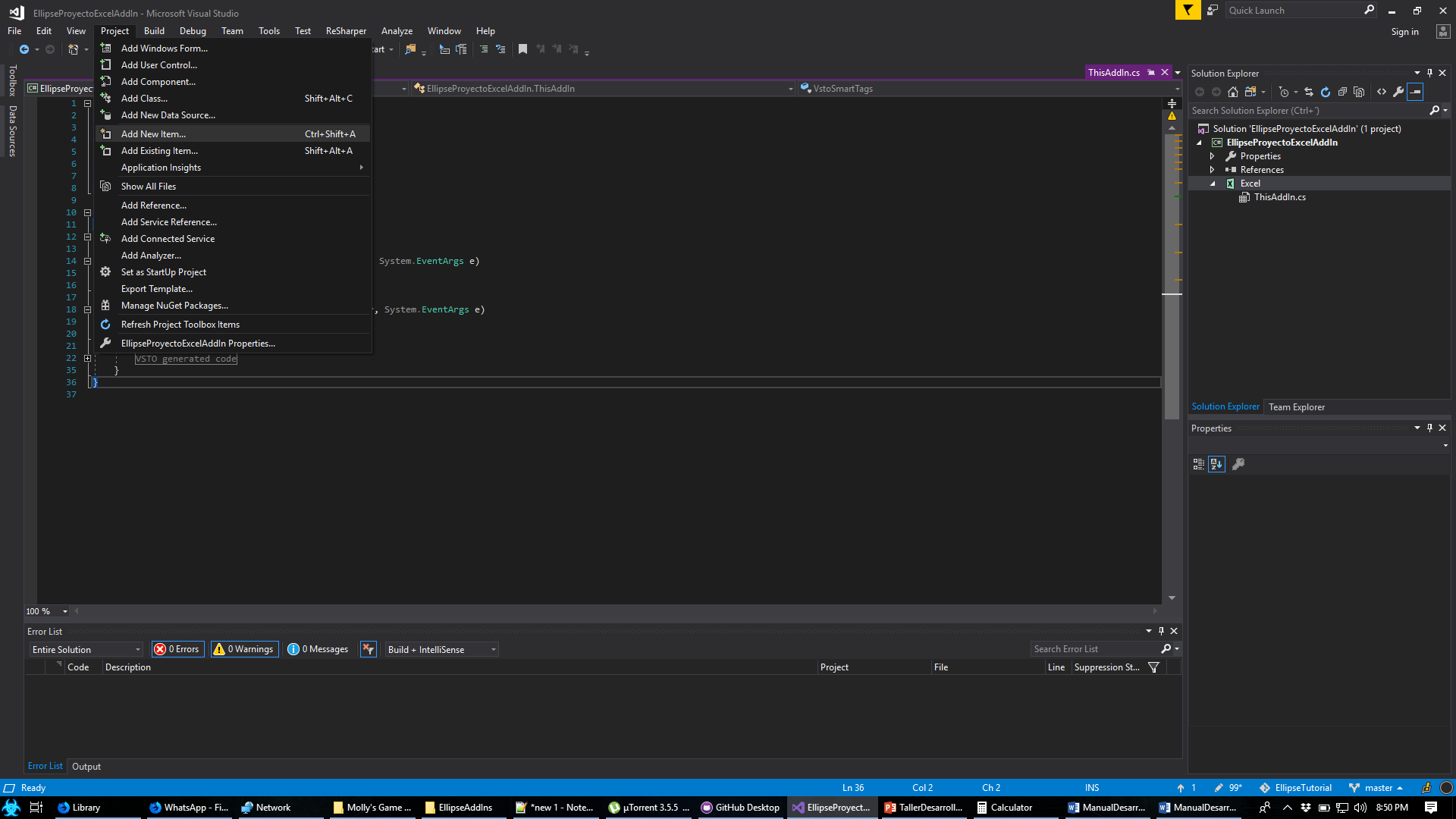
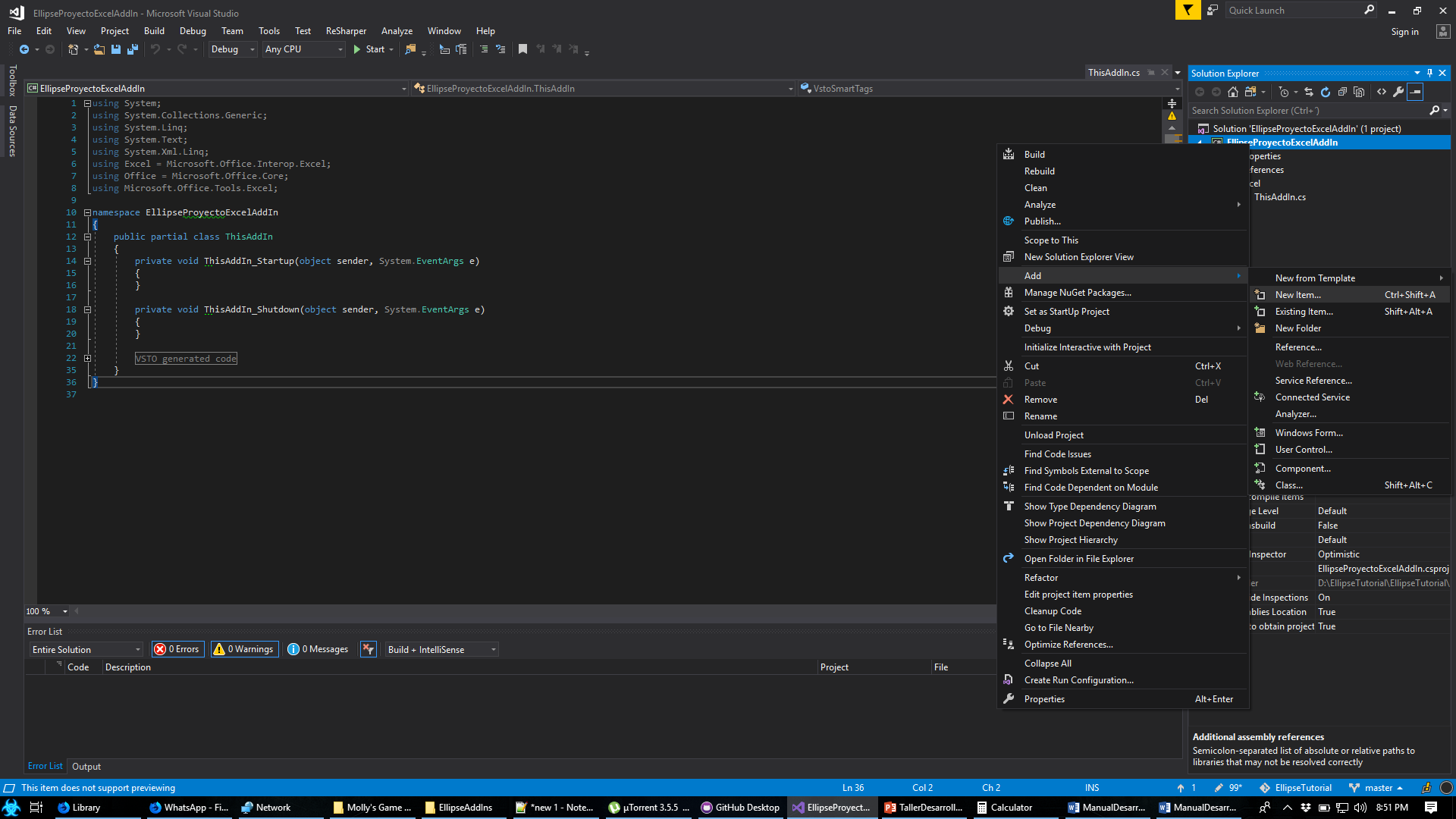


Imagen . Crear archivo de configuración app.config: agregar nuevo

Seleccionamos en la barra lateral izquierda la opción de [4] Visual C#Items -> General y como tipo de archivo [5] Archivo de texto (Text File). Finalmente ingresamos el nombre [6] app.config y hacemos clic en el botón de [7] Agregar (Add)

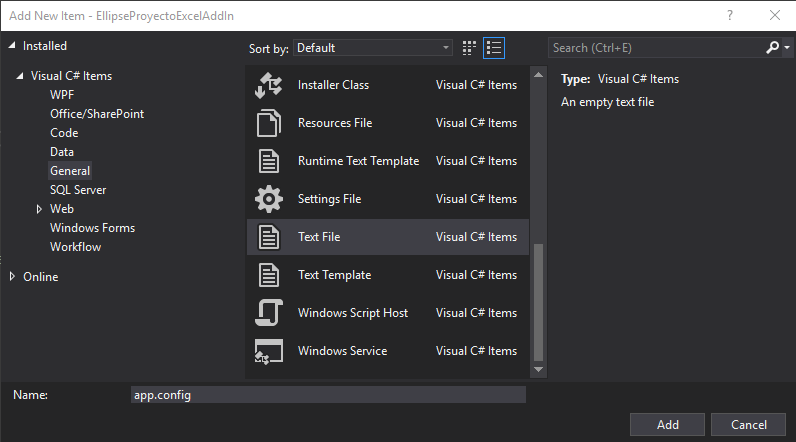


Imagen . Crear archivo de configuración app.config: crear nuevo

En el explorador de soluciones, veremos el nuevo archivo “app.config” creado. Damos doble clic en este para abrirlo y agregamos el contenido mostrado a continuación.

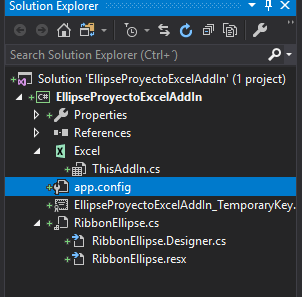


Imagen . Crear archivo de configuración app.config: agregar contenido

Tenga en cuenta los lugares resaltados en amarillo que corresponden a la versión que se ha adicionado al proyecto. Si instala una versión diferente deberá especificarla en estos lugares.

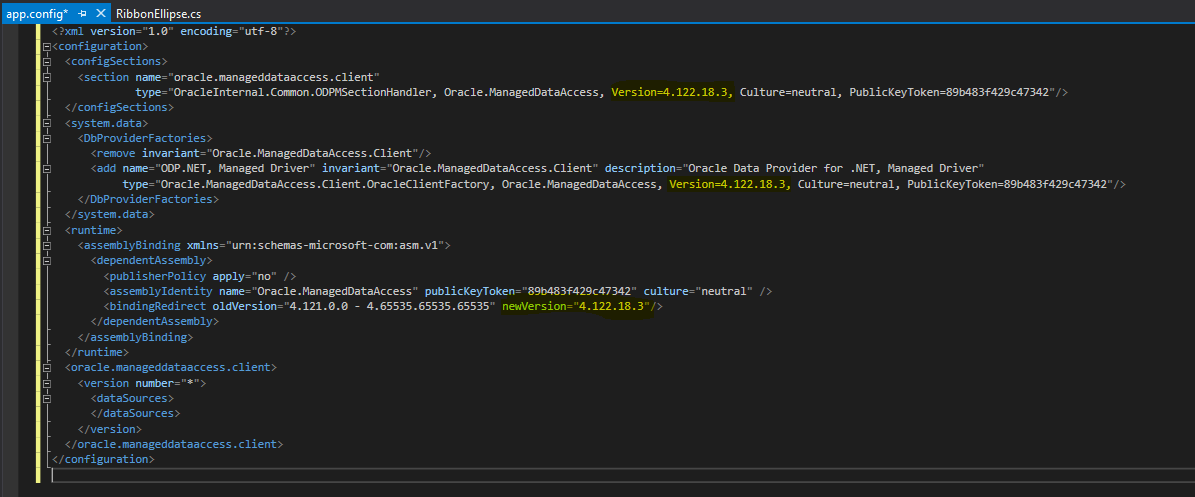


Imagen . Crear archivo de configuración app.config: detalle de contenido

Nota: Junto a este documento encontrará unos archivos anexos con este archivo creado completamente. Podrá copiar su contenido y omitir la creación manual del archivo.

### Agregar referencias TNS/DataSources de acceso a bases de datos

De forma predeterminada, la librería ODP hace uso de las fuentes de datos ingresadas manualmente en el archivo de configuración app.config creado al instalar la librería ODP.net. Puede omitir este paso si hará uso de la librería EllipseCommonsClassLibrary.

Abres el archivo *app.config* dentro del proyecto

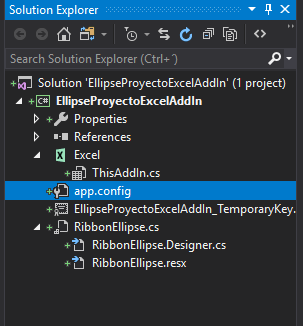


Imagen . Referenciar DataSources: Abrir app.config

Y copias el código dentro de la etiqueta de <oracle.manageddataaccess.client> la información de gestión de acceso del Oracle según la forma de trabajo deseada.

Para TNS:

<settings>

<setting name="TNS\_ADMIN" value="C:\oracle\product\11.2.0\client\network\ADMIN"/>

</settings>

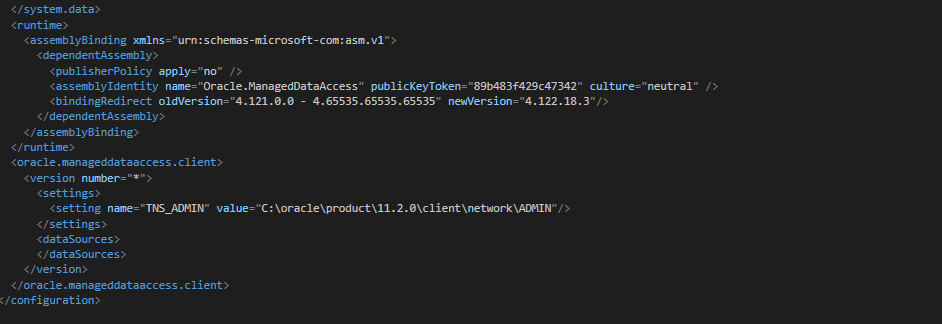


Imagen . Referenciar Bases de Datos: Definiendo la ruta del tnsnames.ora

Para DataSources:

<dataSources>

<dataSource alias="EL8PROD" descriptor="(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=lmndbs09)(PORT=1521))(CONNECT\_DATA=(SERVICE\_NAME=el8prod))) "/>

<dataSource alias="EL8TEST" descriptor="(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=lmndbs05)(PORT=1521))(CONNECT\_DATA=(SERVICE\_NAME=el8test))) "/>

<dataSource alias="EL8DESA" descriptor="(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=lmndbs05)(PORT=1521))(CONNECT\_DATA=(SERVICE\_NAME=el8desa))) "/>

</dataSources>

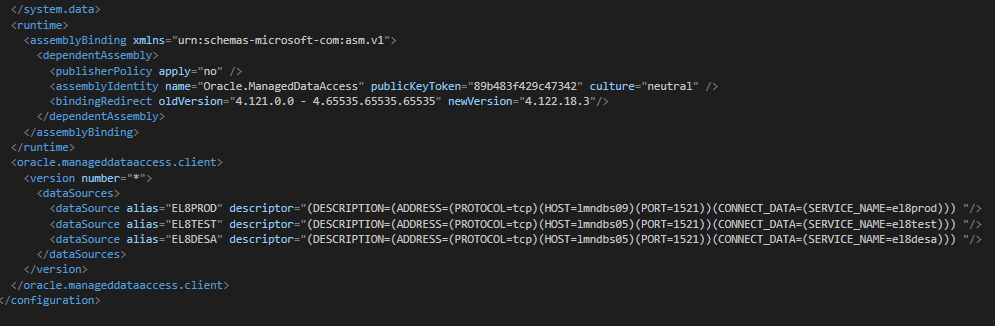


Imagen . Referenciar Bases de Datos: Definiendo por DataSources

Nota:

Hay que tener en cuenta que cada sistema de Oracle tiene su propia configuración de datasources (Ejemplo: El fichero TNSNAMES.ora que suele estar localizado en la ruta indicada en la variable de entorno del equipo según la versión y el sistema de bases de datos instalado). Por lo tanto, para acceder a las bases de datos deberá hacer los ajustes de configuración necesarios según la necesidad de su sistema. Para el ejercicio asumiremos que se utilizarán los DataSources especificados en el archivo app.config.

La librería EllipseCommonsClassLibrary hace uso de su propia configuración para acceder a los DataSources, por lo que si se usa esta librería no es necesario adicionar los DataSources al app.config. Esta configuración crea una variable local de entorno para referenciar la ruta del archivo tnsnames.ora. De forma predeterminada, la ruta corresponde a la establecida en las políticas de TI de Cerrejón (C:\ORACLE\PRODUCT\11.2.0\CLIENT\NETWORK).

### Agregar código de referencia de Oracle Data Manager

Para hacer uso de la librería se deberá adicionar el llamado en los namespaces de la clase que hará uso de ella.

using Oracle.ManagedDataAccess.Client;

Nota: Puede copiar el código del archivo ExecuteQuery.txt adjunto a este Manual para comprobar los resultados del proceso. Tenga presente que probablemente deba agregar referencias en el namespace adicional según lo indique el archivo adjunto.

## AGREGAR REFERENCIA A LIBRERÍAS COMUNES

Las librerías comunes son herramientas desarrolladas por diferentes grupos de desarrolladores, entre ellos la contratista de Hambings SAS, y sirven para facilitar el desarrollo de los AddIns al optimizar y automatizar muchas de las funciones comunes de gestión de hojas de cálculo de Excel, conexión a bases de datos, y acceso a los servicios de Ellipse.

Entre las clases más representativas de la librería común tenemos las siguientes:

**ExcelStyleCells:** con diversos métodos para manejar la lectura y escritura de las hojas de cálculo, manejo de estilo, creación dinámica de tablas, validación de datos de celda, entre otras opciones.

**EllipseFunctions:** con diversos métodos para manejar la conexión a las bases de datos y diferentes entornos de Ellipse, consultar queries, y acceso al servicio de screen service de Ellipse.

**Autenticación de Ellipse:** con los cuadros de diálogos requeridos para solicitar una autenticación con Ellipse y hacer los consumos SOAP respectivos.

**Otras Clases**: la librería cuenta con diferentes clases para el manejo de respuesta de los servicios SOAP, códigos de la tabla 010, superintendencias, grupos de trabajo, entre otros.

### Vincular el proyecto EllipseCommonsClassLibrary

Desplegamos el [1] menú de opciones del proyecto sea desde el menú principal o haciendo clic derecho en el explorador de soluciones, y seleccionamos “[2] Agregar (Add) > [3] Referencia (Reference)”

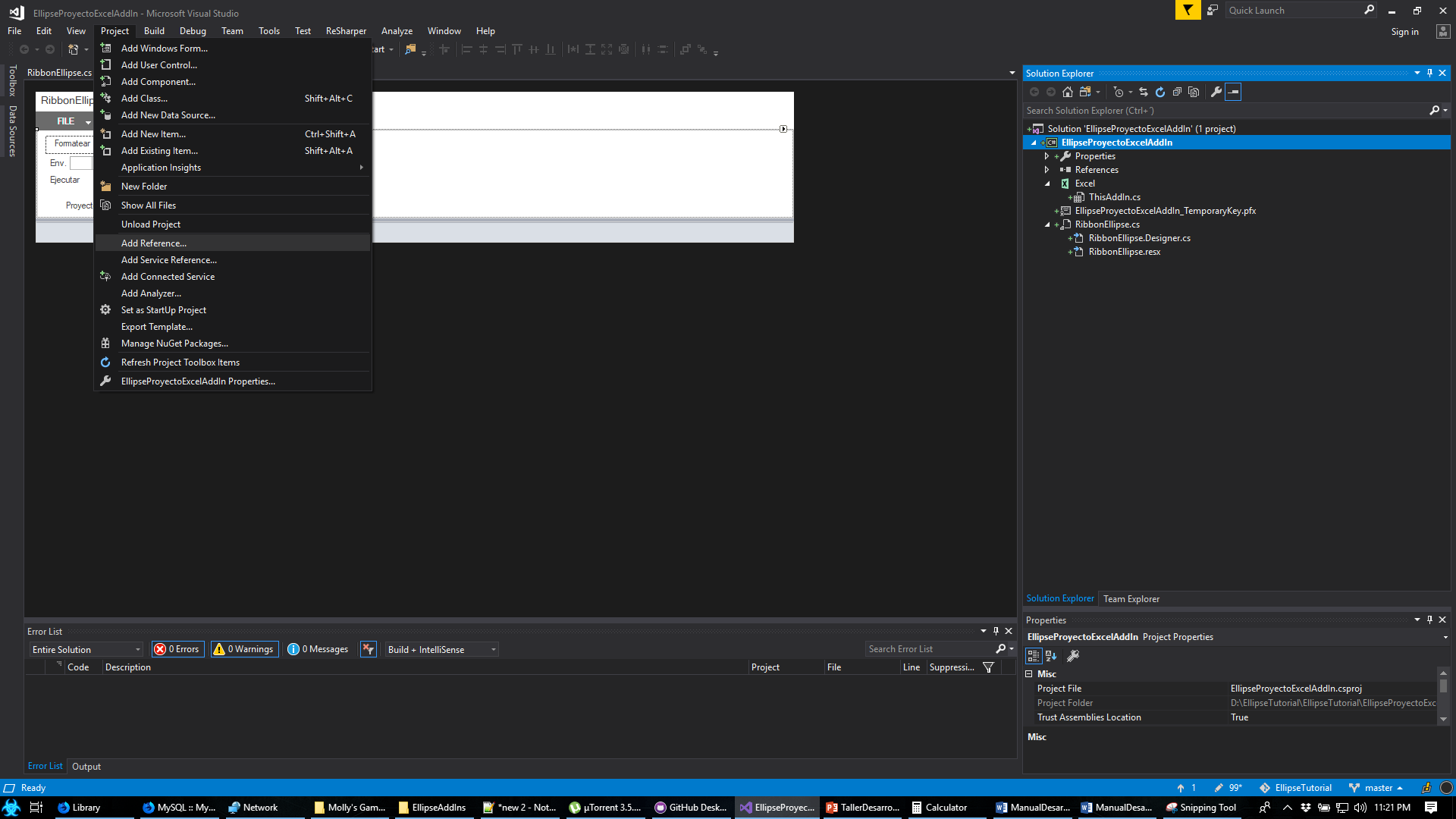
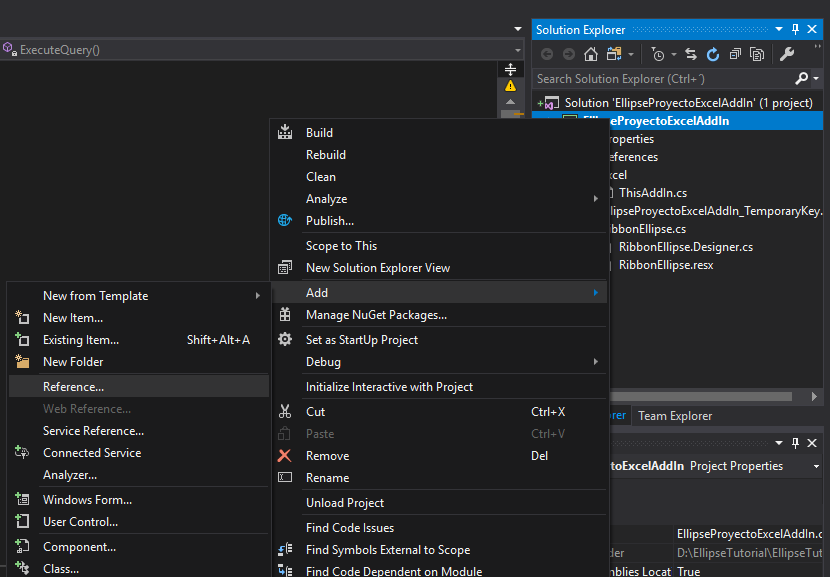


Imagen . Adicionar Librerías Comunes: Agregar Nueva Librería

En la barra lateral izquierda seleccionamos la opción de [4] Búsqueda (Browse) y nuevamente en el botón de [5] Búsqueda (Browse). Ubicamos la ruta donde se encuentra el archivo de librería con la versión correspondiente, seleccionamos [6] Oracle.ManagedDataAccess.dll y hacemos clic en el botón de [7] Añadir (Add).

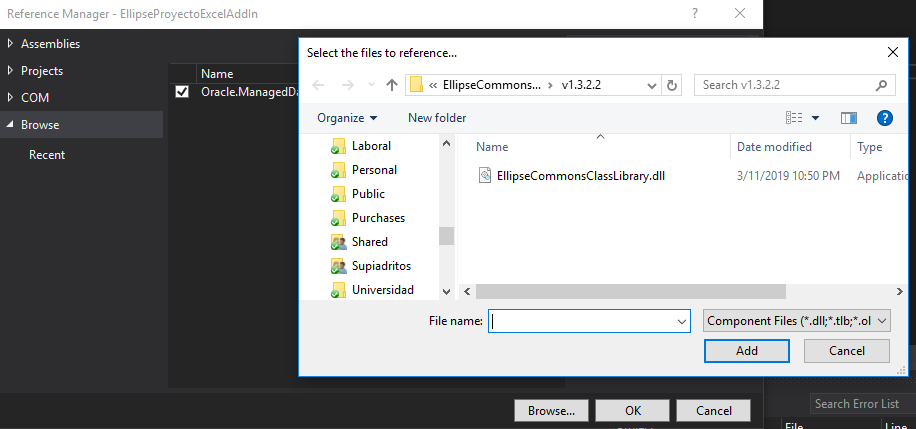


Imagen . Adicionar Librerías Comunes: Agregar EllipseCommonsClassLibrary

Nos aseguramos que el cuadro de selección [8] esté activo en el listado y finalizamos haciendo clic en el botón de [9] OK

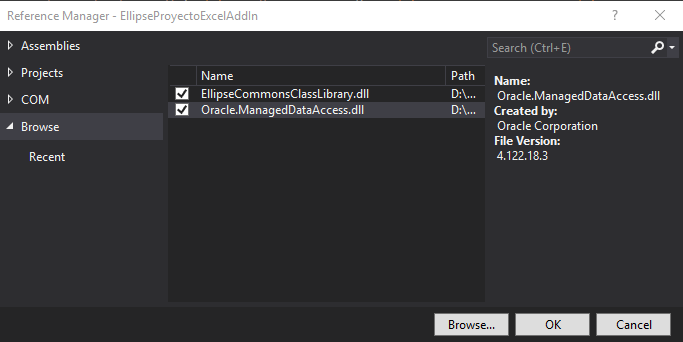


Imagen . Adicionar Librerías Comunes: Confirmar Librería Común

*Repetir el numeral para cada una de las librerías comunes que desee añadir al proyecto.*

### Agregar código de referencias de Librerías Commons

Para hacer uso de la librería se deberá adicionar las siguientes sentencias.

Namespaces:

using EllipseCommonsClassLibrary;

using EllipseCommonsClassLibrary.Classes;

using EllipseCommonsClassLibrary.Connections;

using Excel = Microsoft.Office.Interop.Excel;

Definificación de Variables:

ExcelStyleCells \_cells;

EllipseFunctions \_eFunctions = new EllipseFunctions();

private Excel.Application \_excelApp;

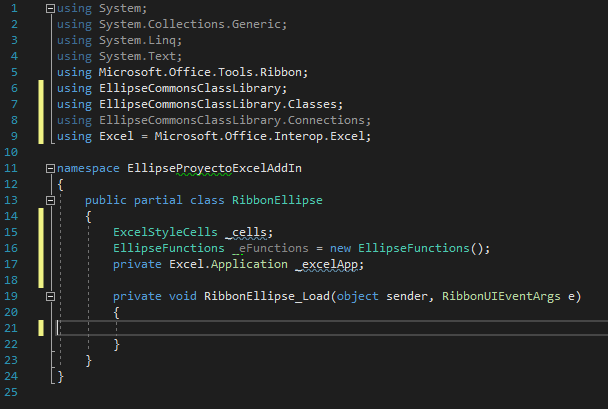


Imagen . Agregar código de referencia a Librería Commons

Adicionamos el siguiente código al método de iniciación de la Cinta de Opciones (Ribbon). Estas sentencias poblarán la información de la lista desplegable de entornos disponibles.

\_excelApp = Globals.ThisAddIn.Application;

var environments = Environments.GetEnvironmentList();

foreach (var env in environments)

{

var item = Factory.CreateRibbonDropDownItem();

item.Label = env;

drpEnviroment.Items.Add(item);

}

Nota: Según la configuración de la librería se hará referencia al archivo de entornos XML donde está disponible la información de servicios de Ellipse.

Ruta Predeterminada: [\\lmnoas02\SideLine\EllipsePopups\Ellipse8\](file:///\\lmnoas02\SideLine\EllipsePopups\Ellipse8\)

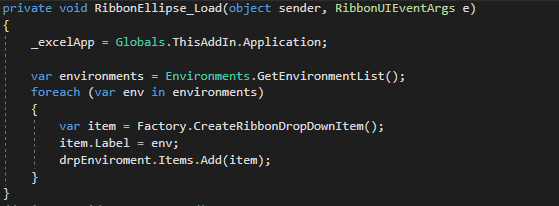


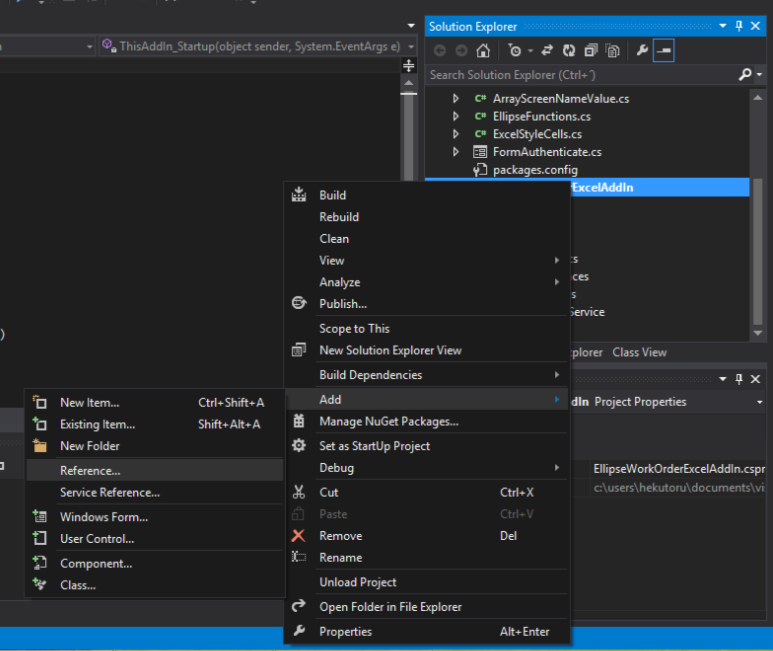
Imagen Agregar código de referencia a Librería Commons: Código de Iniciación

Nota: Puede copiar el código del archivo ExecuteQueryCommons.txt adjunto a este Manual para comprobar los resultados del proceso. Tenga presente que probablemente deba agregar referencias en el namespace adicional según lo indique el archivo adjunto.

## AGREGAR CABECERA DE CONSUMO SOAP

### Agregar referencia a librería de consumo SOAP

Desplegamos el [1] menú de nuestro proyecto y seleccionamos la opción “*[2] Add… > Reference…”*



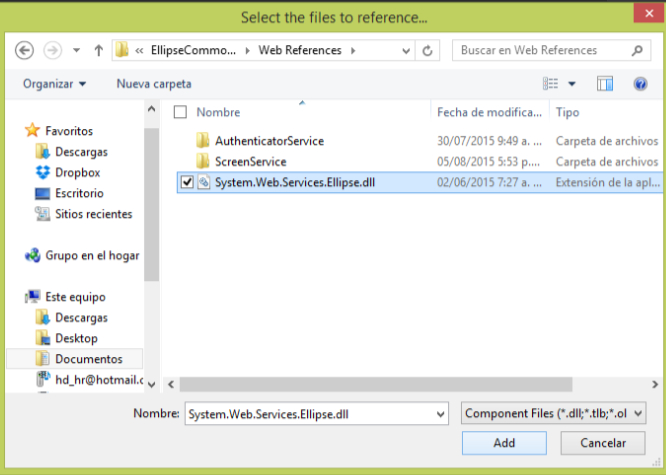
**1**

**2**

**3**

Imagen 14: Agregar Encabezado SOAP: Adicionar referencia

Buscamos la librería dentro del proyecto commons "...\EllipseCommonsExcelAddIn\Web References" y [4] seleccionamos el archivo System.Web.Services.Ellipse.dll. Finalmente le damos al [5] botón Add



**4**

**5**

Imagen 15: Agregar Encabezado SOAP: Seleccionar librería

### Agregar código de referencia de encabezado SOAP en el proyecto

Se añade el siguiente código a la clase principal del desarrollo del proyecto (ver numeral 5.4)

using System.Web.Services.Ellipse;

### Adicionar referencia a app.config

Adicionamos el siguiente código al archivo app.config dentro de las etiquetas de <configuration></configuration>

<system.web>

<webServices>

<soapExtensionTypes>

<add type="System.Web.Services.Ellipse.AuthenticationExtension, System.Web.Services.Ellipse" priority="1"/>

</soapExtensionTypes>

</webServices>

</system.web>

El resultado es como se muestra en la siguiente imagen (ver imagen 16)

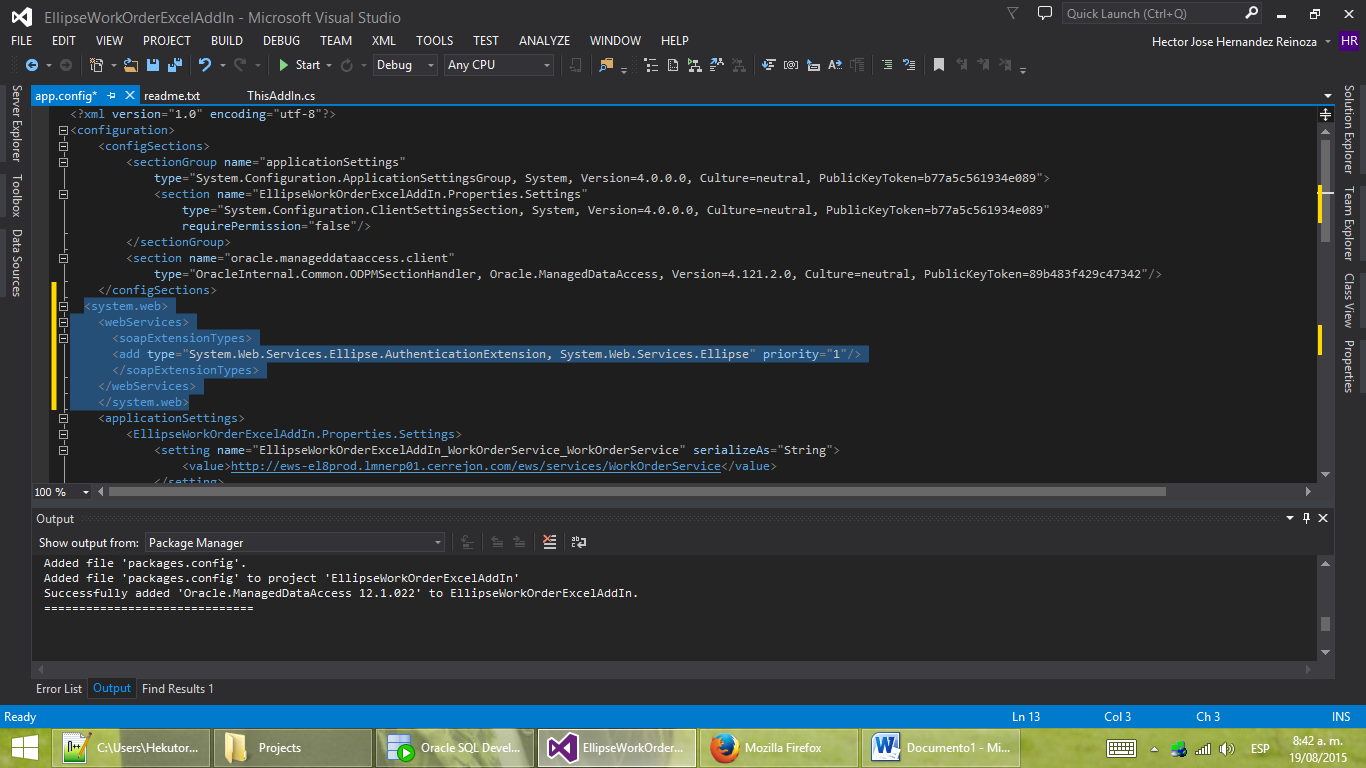


Imagen 16: Agregar Encabezado SOAP: Adicionar información a app.config

## AGREGAR REFERENCIAS WEB Y CUALQUIER OTRA REFERENCIA REQUERIDA

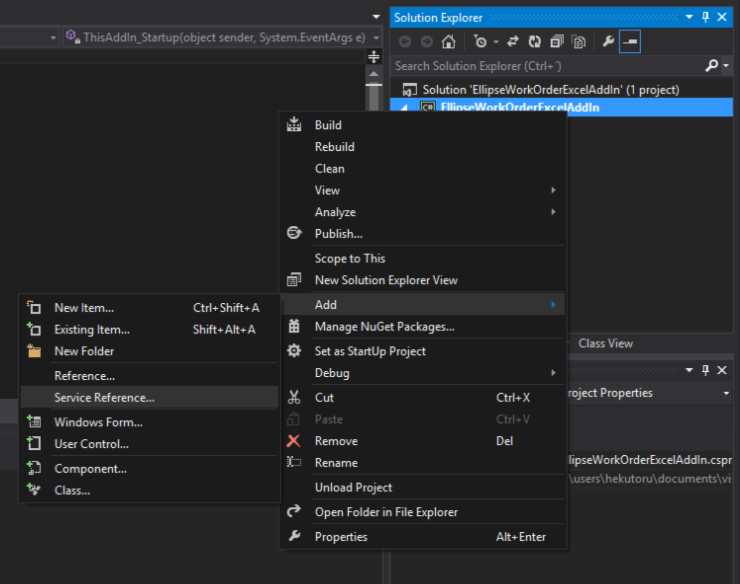
### Agregar código de referencia del Screen Service

Se añade el siguiente código a la clase principal del desarrollo del proyecto (ver numeral 5.4)

using Screen = EllipseCommonsClassLibrary.ScreenService;

### Agregar Web Service de Ellipse

[1] Desplegamos el menú de opciones del proyecto de la solución y seleccionamos *“[2] Add > [3] Service Reference …”*



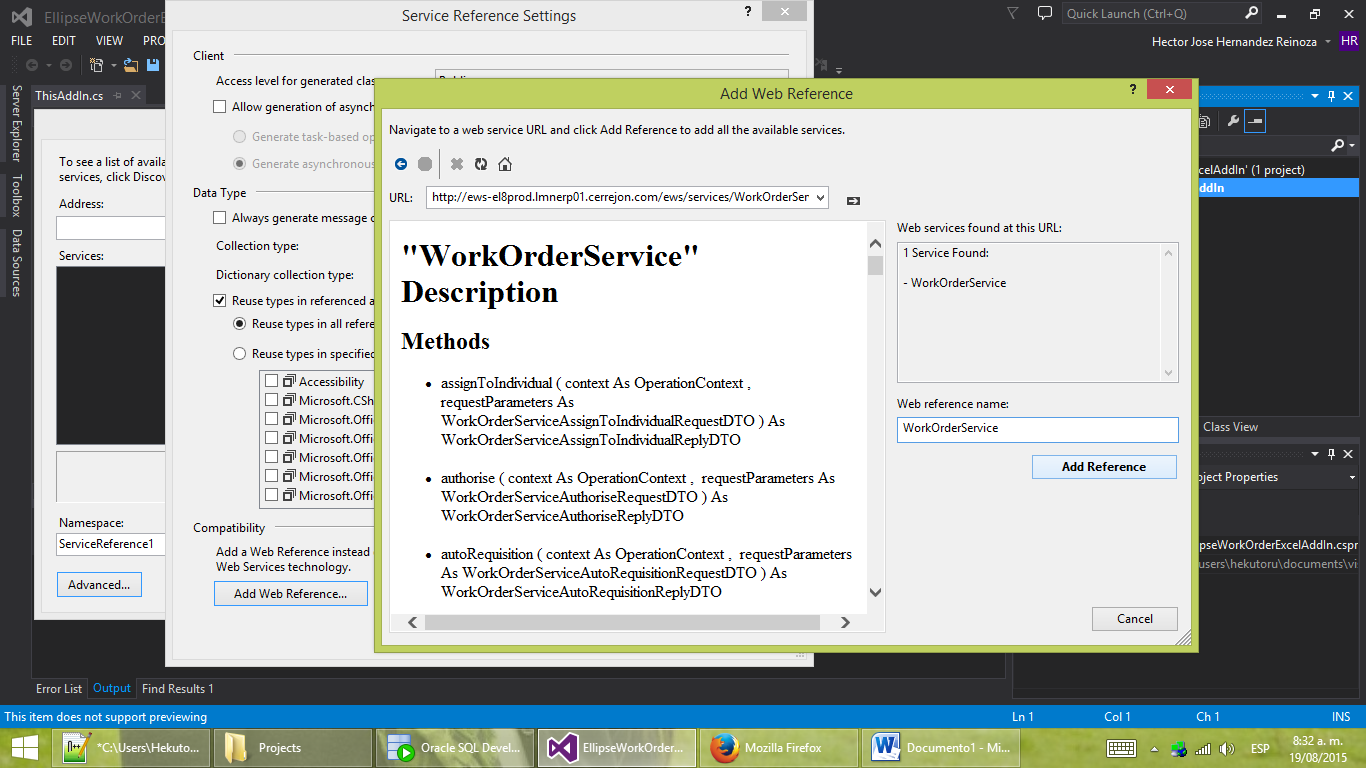
**1**

**2**

**3**

Imagen 3: Agregar Referencias Web: Agregar referencia a servicio

[4] Seleccionamos el botón *“Advanced…”*, para más opciones avanzadas y luego [5] el botón *“Add Web Reference…”*. [6] Copiamos la URL del servicio web que deseamos añadir en el cuadro de URL y [7] le damos clic en el botón de enviar. [8] Finalmente le damos nombre al servicio y seleccionamos [9] el botón de *“Add Reference”*



**4**

**5**

**6**

**7**

**8**

**9**

Imagen 4: Agregar Referencias Web: Especificar Web Service de Ellipse

*El servicio de ScreenService no necesita ser referenciado. Si el desarrollo va a hacer uso de este servicio, puede invocarlo directamente del Proyecto EllipseCommonsClassLibrary descrito en el siguiente numeral.*

*Si no va a adicionar ningún servicio, deberá agregar de forma manual la referencia de System.Web.Services (este servicio se agrega automáticamente cuando se agrega cualquier servicio web)*

### Agregar referencia de Web Service de Ellipse en el Proyecto

Se añade el siguiente código a la clase principal del desarrollo del proyecto (ver numeral 5.4)

using EllipseProyectoExcelAddIn.ProyectoService;

## OBSERVACIONES ADICIONALES DE DESARROLLO

A continuación se dejan algunas observaciones adicionales para el desarrollo

### Autenticación

//Se realiza la autenticación

\_frmAuth.StartPosition = FormStartPosition.CenterScreen;

\_frmAuth.SelectedEnviroment = drpEnviroment.SelectedItem.Label;

if (\_frmAuth.ShowDialog() != DialogResult.OK) return;// si no se autentica, se cancela el proceso

//se adiciona el código de la acción a realizar en el proceso

Ver imagen 27

### Uso del servicio seleccionado

* + 1. Screen Service

//Instanciar el Contexto de Operación

var opSheet = new OperationContext

{

district = \_frmAuth.EllipseDsct,

position = \_frmAuth.EllipsePost,

maxInstances = 100,

maxInstancesSpecified = true,

returnWarnings = \_eFunctions.DebugWarnings,

returnWarningsSpecified = true

};

//Instanciar el SOAP

ClientConversation.authenticate(\_frmAuth.EllipseUser, \_frmAuth.EllipsePswd);

//Proceso del screen

var proxySheet = new Screen.ScreenService();

var requestSheet = new Screen.ScreenSubmitRequestDTO();

//Se define el ambiente del Dropdown de Enviroment

var urlService = \_eFunctions.GetServicesUrl(drpEnviroment.SelectedItem.Label);

proxySheet.Url = urlService + "/ScreenService";

//ejecutamos el programa y leemos la respuesta si se necesita en el replySheet

var replySheet = proxySheet.executeScreen(opSheet, "MSO435");

//validamos el ingreso al programa

if (replySheet.mapName != "MSM435A" || \_excelApp.ActiveWorkbook.ActiveSheet.Name != SheetName01)

return "ERROR:" + "No se pudo establecer comunicación con el servicio";

//arreglo para los campos del screen

var arrayFields = new ArrayScreenNameValue();

//se adicionan los campos que se vayan a enviar

arrayFields.Add("OPTION1I", "1");

arrayFields.Add("MODEL\_CODE1I", modelCode);

arrayFields.Add("STAT\_DATE1I", modelDate);

arrayFields.Add("SHIFT1I", modelShift);

//se envía el screen

requestSheet.screenFields = arrayFields.ToArray();

requestSheet.screenKey = "1";

replySheet = proxySheet.submit(opSheet, requestSheet);

//manejo de opciones para el tipo de respuesta

if (replySheet != null

&& replySheet.mapName == "MSM435B")

&& !\_eFunctions.CheckReplyError(replySheet)

&& !\_eFunctions.CheckReplyWarning(replySheet)

&& replySheet.functionKeys = "XMIT-Confirm")

{

//si necesitas obtener los campso del reply y trabajar con ellos

var replyFields = new ArrayScreenNameValue(replySheet.screenFields);

bool isEmpty = replyFields.getField("WO\_PROJ1I" + i).value.Equals("");

}

*Ver imagen 27 y 28*

* + 1. Otros servicios (MSE)

var opSheet = new WorkOrderService.OperationContext();

opSheet.district = \_frmAuth.EllipseDsct;

opSheet.position = \_frmAuth.EllipsePost;

opSheet.maxInstances = 100;

opSheet.maxInstancesSpecified = true;

opSheet.returnWarnings = \_eFunctions.DebugWarnings;

opSheet.returnWarningsSpecified = true;

var proxyWo = new WorkOrderService.WorkOrderService();//ejecuta las acciones del servicio

var requestWo = new WorkOrderServiceCreateRequestDTO();

proxyWo.Url = urlService + "/WorkOrder";

//Se cargan los parámetros de la solicitud

requestWo.districtCode = "ICOR";

requestWo.workGroup = "MTOLOC";

//se envía la acción

var replyWO = proxyWo.modify(opContext, requestWo);

//se analiza el reply según la acción

//se hacen las acciones pertinentes

*Ver imagen 29*

### Ejecución de Querys

//EJECUCIÓN DE QUERIES

//Estableces el servidor de base de datos con alguna de las siguientes tres opciones

\_eFunctions.SetDBSettings(urlService);

\_eFunctions.SetDBSettings(dbname, dbuser, dbpass);

\_eFunctions.SetDBSettings(dbname, dbuser, dbpass, dblink, dbreference);

var dataReader = ef.GetQueryResult(sqlQuery);

if (dataReader == null || dataReader.IsClosed || !dataReader.HasRows) return null;

while (dataReader.Read())

{

var workOrder = dataReader["WORK\_ORDER"].ToString().Trim();

}

Ver imagen 30

### Control de Errores y Debugging

//CONTROL DE ERRORES Y DEBUGGING

//debugging de queries

if(\_eFunctions.DebugQueries)

\_cells.getCell("L1").Value = sqlQuery;

//GESTIÓN DE EXCEPCIONES

try

{

if (\_excelApp.ActiveWorkbook.ActiveSheet.Name != SheetName01 + LabourSheetTypeConstants.Default)

throw new Exception("La hoja seleccionada no coincide con el modelo requerido");

if (drpEnviroment.Label == null || drpEnviroment.Label.Equals(""))

throw new ArgumentException("Seleccione un ambiente válido");

}

catch (Exception ex)

{

//Despliega una alerta de error o realiza la acción según se requiera

MessageBox.Show(ex.Message, @"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

//Genera un log del error. Si \_eFunctions.DebugErrors es true, siempre desplegará un mensaje de error

ErrorLogger.LogError("RibbonEllipse:startLabourCostLoad()", "\n\rMessage:" + ex.Message + "\n\rSource:" + ex.Source + "\n\rStackTrace:" + ex.StackTrace, \_eFunctions.DebugErrors);

}

Ver imagen 30

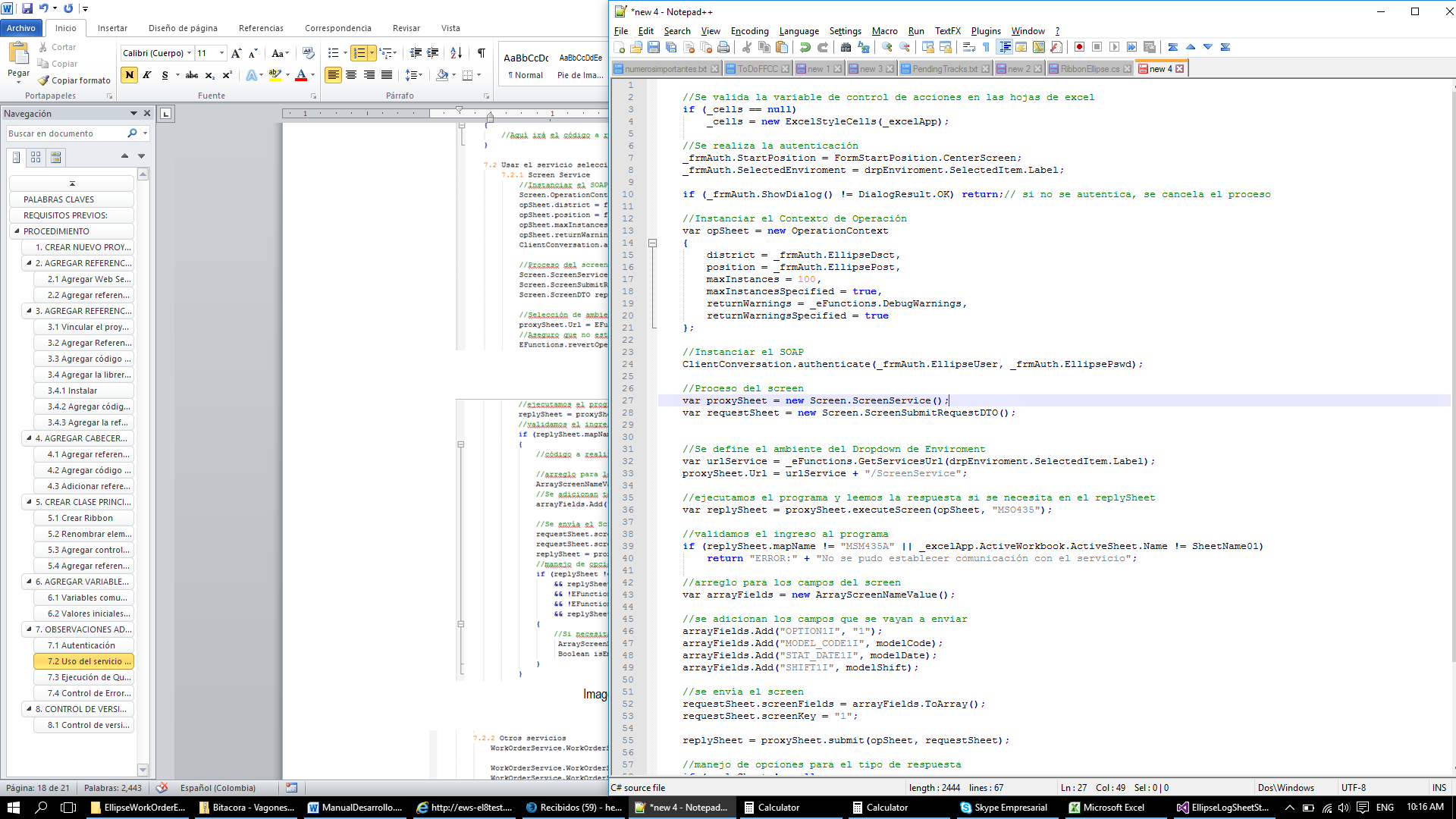


Imagen 27: Observaciones Adicionales: Autenticación y uso de servicios

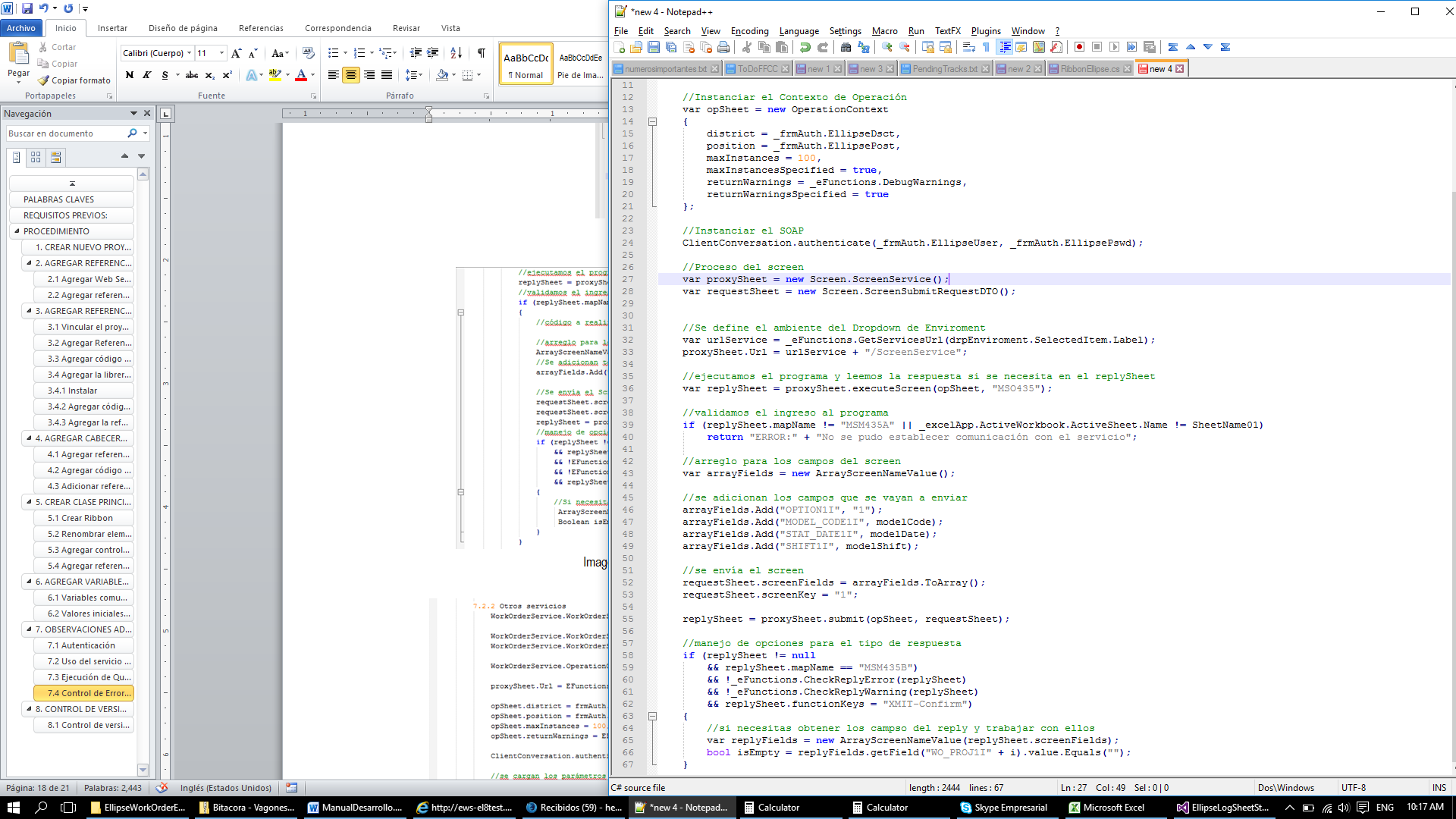


Imagen 28: Observaciones Adicionales: Continuación de uso de servicios y ScreenService

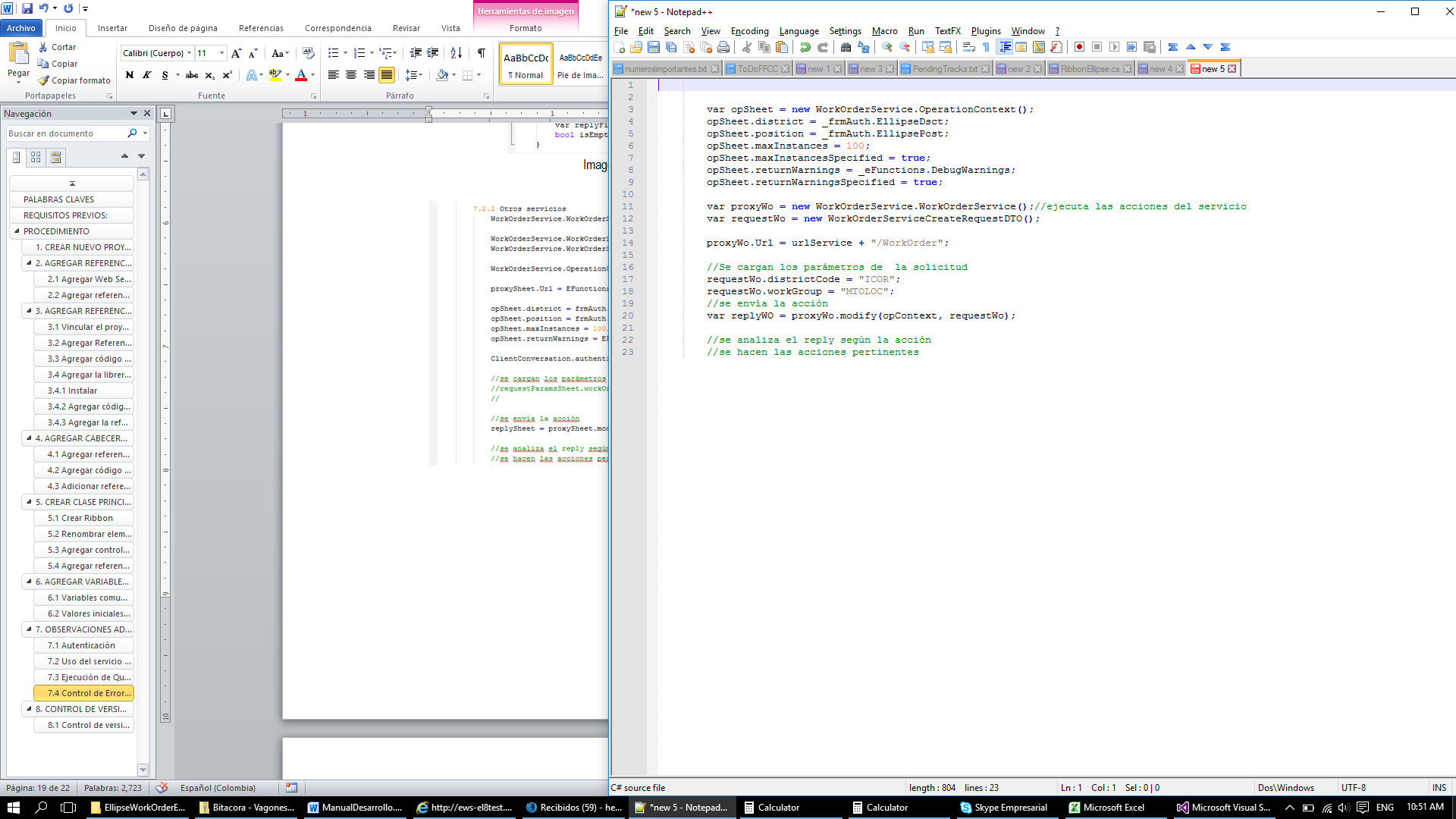


Imagen 29: Observaciones Adicionales: Uso de otros servicios y MSE

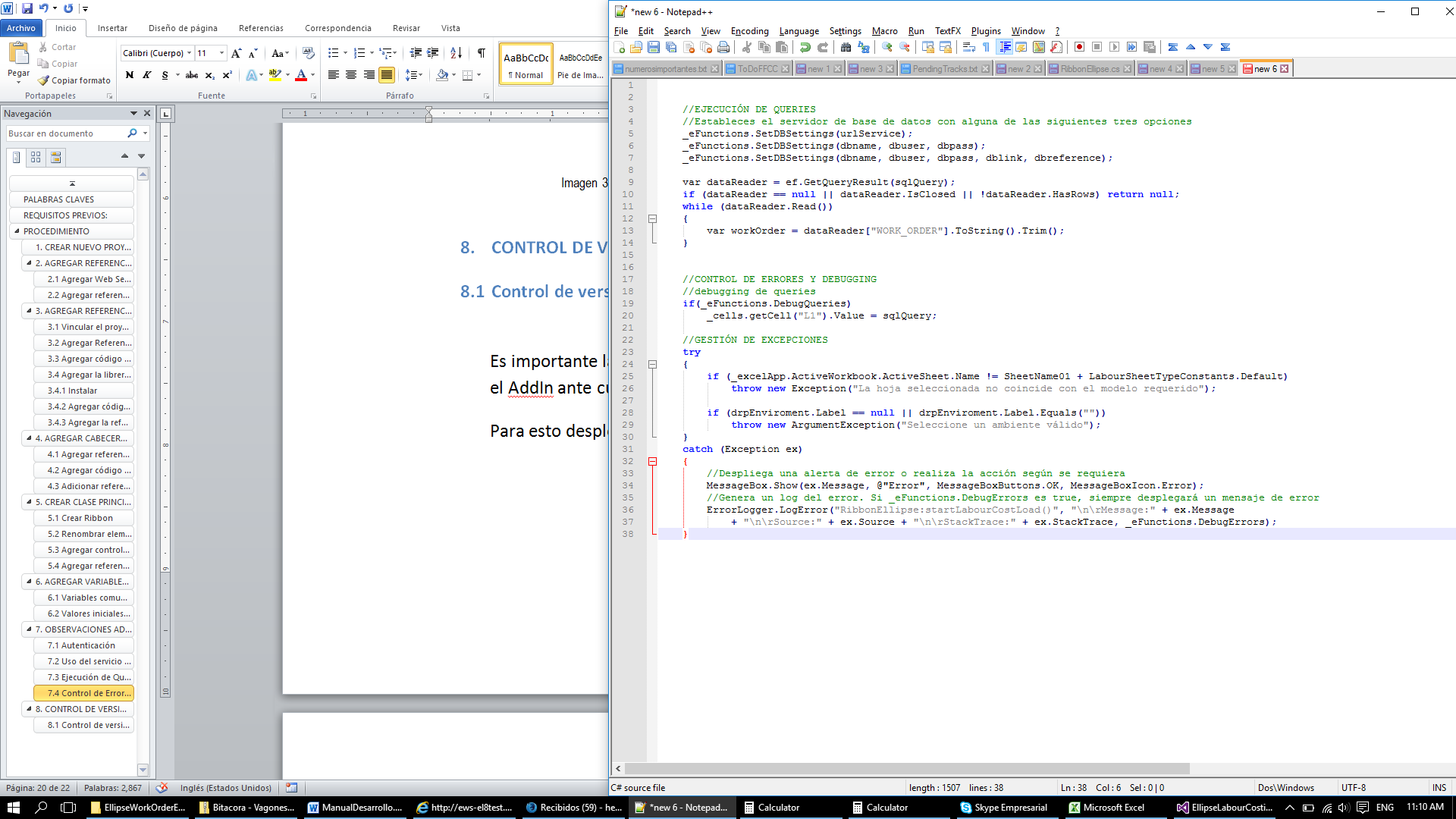


Imagen 30: Observaciones Adicionales: Uso, ejecución de Querys y control de errores y debugging

### Validaciones de Campo

Para utilizar validaciones de campo en las hojas de Excel utilizamos una variable global con el nombre de la hoja de Excel donde vamos a tener el listado de validaciones.

private const string ValidationSheetName = "ValidationSheetWorkOrder";

*Se opta por tener el listado de validaciones en una hoja de Excel porque de hacerlo dinámicamente, al cerrar y reabrir el documento se perderán los valores en la fórmula de validación.*

Hacemos uso de algunas de las tres formas para establecer la validación:

1. SetValidationList(Range targetRange, List<string> validationValues, bool showError = true)

Establece en las celdas especificadas los valores de validación de forma dinámica. *No crea datos en la hoja de validación, ni usa la hoja de validación*

1. SetValidationList(Range targetRange, List<string> validationValues, string validationSheetName, int validationColumnIndex, bool showError = true)

Establece en las celdas especificadas los valores de validación escribiéndolos en la hoja de validación según la columna indicada. *Crea datos en la hoja de validación según la lista de datos ingresada*

1. SetValidationList(Range targetRange, string validationSheetName, int validationColumnIndex, bool showError = true)

Establece en las celdas especificadas los valores de validación tomándolos de la hoja de validación según la columna indicada. *Hace uso de los datos ya existentes en la hoja de validación*

targetRange: rango de celdas objetivo que tendrán la validación de datos

validationValues: lista de datos que se utilizarán como muestra de validación

validationSheetName: nombre de la hoja de Excel donde se agregarán u obtendrán los datos de validación

validationColumnIndex: índice de la columna de la hoja de validación de Excel donde se agregarán u obtendrán los datos de validación

showError: indica si se mostrará el mensaje de alerta de error cuando se ingresen datos que no correspondan en la celda

Obtenemos el valor del campo y lo utilizamos según como necesitemos:

//obtengo el valor de la celda

var codeComp = \_cells.GetNullIfTrimmedEmpty(\_cells.GetCell(15, i).Value);

//obtengo solo el código sin la descripción (“cod – description”)

if (codeComp != null && codeComp.Contains(" - "))

codeComp = codeComp.Substring(0, codeComp.IndexOf(" - ", StringComparison.Ordinal));

A continuación algunos ejemplos:

//EJEMPLO CON TABLA DE CÓDIGOS 010

//obtengo el listado de códigos

var itemList1 = \_eFunctions.GetItemCodes("SC");

//creo la variable de lista con código - descripción

var compCodeList = itemList1.Select(item => item.code + " - " + item.description).ToList();

//asigno los datos a la hoja de validación y la validación a la celda respectiva

\_cells.SetValidationList(\_cells.GetCell(15, i), compCodeList, ValidationSheetName, 2);

//para hacer uso de lo seleccionado. Obtengo el valor de la celda

var codeComp = \_cells.GetNullIfTrimmedEmpty(\_cells.GetCell(15, i).Value);

//elimino la descripción y quedo solo con el código

if (codeComp != null && codeComp.Contains(" - "))

codeComp = codeComp.Substring(0, codeComp.IndexOf(" - ", StringComparison.Ordinal));

//uso la variable codeComp según necesite

//EJEMPLO CON DISTRITO Y GRUPOS DE TRABAJO

//Adicionar validaciones

\_cells.SetValidationList(\_cells.GetCell("B3"), DistrictConstants.GetDistrictList(), ValidationSheetName, 1);

\_cells.SetValidationList(\_cells.GetCell("B4"), GroupConstants.GetWorkGroupList().Select(g => g.Name).ToList(), ValidationSheetName, 2);

var district = \_cells.GetNullIfTrimmedEmpty(\_cells.GetCell("B3").Value);

var workGroup = \_cells.GetNullIfTrimmedEmpty(\_cells.GetCell("B4").Value);

//uso la variable district y workGroup según necesite

//EJEMPLO CON LISTAS CORTAS DINÁMICAS

\_cells.SetValidationList(\_cells.GetCell(8, i), new List<string> { "P - Project", "W - WorkOrder" }, ValidationSheetName, 1);

var referenceType = \_cells.GetNullIfTrimmedEmpty(\_cells.GetCell(8, i).Value);

//elimino la descripción y quedo solo con el código

if (referenceType != null && referenceType.Contains(" - "))

referenceType = referenceType.Substring(0, referenceType.IndexOf(" - ", StringComparison.Ordinal));

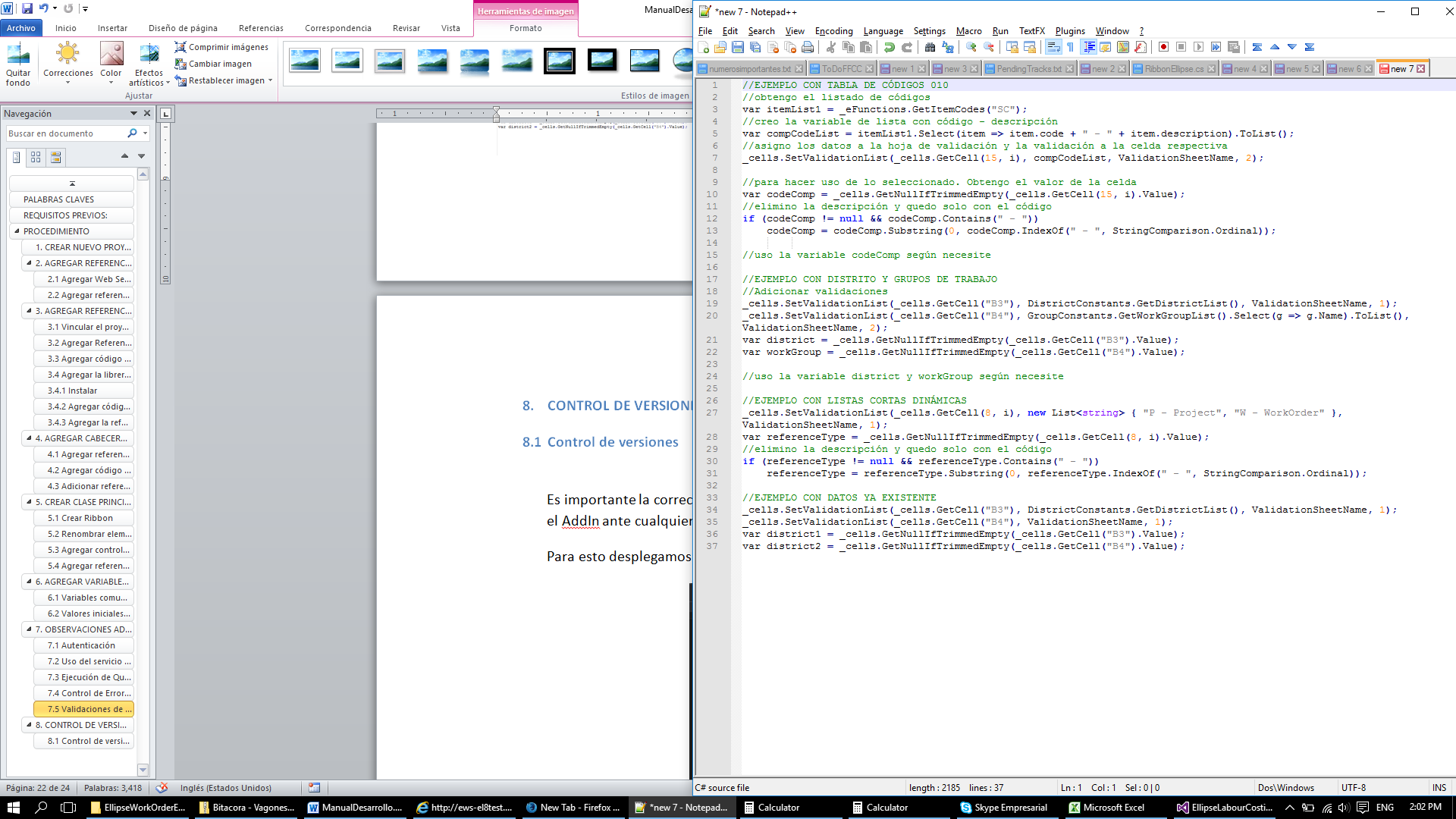
//EJEMPLO CON DATOS YA EXISTENTE

\_cells.SetValidationList(\_cells.GetCell("B3"), DistrictConstants.GetDistrictList(), ValidationSheetName, 1);

\_cells.SetValidationList(\_cells.GetCell("B4"), ValidationSheetName, 1);

var district1 = \_cells.GetNullIfTrimmedEmpty(\_cells.GetCell("B3").Value);

var district2 = \_cells.GetNullIfTrimmedEmpty(\_cells.GetCell("B4").Value);



Imágen 31. Ejemplos de uso de validaciones

### Gestión de múltiples hojas de Excel

Para la gestión las hojas de Excel se utiliza la clase *ExcelStyleCells.cs*, por lo que hay que conocer algunos elementos básicos de la misma.

La declaración de una variable de esta clase está sujeta a la ejecución de la aplicación y la hoja activa de la misma. Utiliza un apuntador que establece la hoja con la que se definió al momento de la creación. Si no se definió ninguna hoja, la hoja definida será la hoja principal.

Si hace uso de la instancia global, deberá corroborar que haya sido inicializada en cada método que use. Puede hacer uso de cualquiera de las opciones de inicialización más adelante descritas.

if (\_cells == null)

\_cells = new ExcelStyleCells(\_excelApp);

1. Instanciar sin definir la hoja principal

var \_cells = new ExcelStyleCells (Application excelApp,

bool alwaysActiveSheet = true);

Al definir la variable de esta forma, siempre que se haga referencia a *\_cells* se estará haciendo referencia a la hoja activa de Excel dentro del proceso. La hoja base será la hoja que estaba activa al momento de la creación de la variable.

1. Instanciar definiendo la hoja principal

var \_cells = ExcelStyleCells(Application excelApp, string sheetName);

Al definir la variable de esta forma, siempre que se haga referencia a \_cells se estará haciendo referencia a la hoja base, la hoja con el nombre *sheetName* dado inicialmente, sin importar si es la hoja activa o no.

Existen otros métodos para variar este comportamiento después que la variable haya sido instanciada.

\_cells.SetActiveSheet();

Establece como hoja base la hoja activa que tenga la aplicación en el momento. No modifica el comportamiento de referenciación

\_cells.SetActiveSheet(string sheetName);

Establece como hoja base la hoja ingresada. No modifica el comportamiento de referenciación

\_cells.SetAlwaysActiveSheet(bool value);

Establece el comportamiento de referenciación. *True* para que la referencia siempre sea la hoja activa y *False* para que sea la hoja base.

\_cells.ToggleAlwaysActiveSheet();

Cambia el estado del comportamiento de referenciación. Si estaba activo lo desactiva y viceversa.

### Manejo de hilos de procesos

Para el desarrollo de algunos loaders puede ser requerido el manejo de hilos de proceso con el fin de evitar que la aplicación entre en estado de *no responde* hasta que se termine el proceso solicitado.

VSTO tiene algunas restricciones en cuanto a desempeño y manejo de estos hilos de procesos, por lo que si lo requiere puede investigar más al respecto. A continuación hacemos uso de los hilos de procesos tomando en cuenta algunas recomendaciones en la implementación de los mismos para VSTO.

Definición:

Hacemos uso de la librería del sistema para los hilos del proceso.

using System.Threading;

Dentro de nuestra clase principal (RibbonEllipse) definiríamos la variable que vamos a utilizar para la gestión de los hilos del proceso. *Si requiere más de un hilo de proceso simultáneo puede crear más de una variable.*

private Thread \_thread;

Iniciar un hilo de proceso:

Para hacer uso de un hilo de proceso validamos que este no haya sido iniciado previamente y que todavía esté corriendo. *De no hacer esta validación se podrían crear incontrolables números de hilos de proceso lo cual podría llevar a errores en los procesos requeridos*. Si ya hay un hilo iniciado y no se ha terminado, bien podría generar una excepción o simplemente no hacer nada.

*Generar Excepción:*

if (\_thread != null && \_thread.IsAlive)

throw new Exception("Ya hay un proceso ejecución. Espere que el proceso termine e intente nuevamente");

*No hacer nada:*

if (\_thread != null && \_thread.IsAlive) return;

Si no hay ningún hilo activo se procede a continuar. Para esto inicializamos la variable que definimos para nuestro hilo y le asignamos el nombre del método que deseamos que lleve a cabo de nuestro hilo (en el ejemplo *CreateWoList*). La siguiente línea de ejecución corresponde a una recomendación de manejo de hilos en VSTO.

\_thread = new Thread(CreateWoList);

\_thread.SetApartmentState(ApartmentState.STA);

Finalmente, invocamos la ejecución del hijo donde se requiera.

\_thread.Start();

*Se recomiendo el uso del método de clase “\_cells.SetCursorWait();” dentro del método que ejecutará el Thread para indicar al usuario mediante la animación del cursor que se está ejecutando un proceso en segundo plano. Recuerde que deberá garantizar que \_cells no sea nulo.*

Detener un hilo de proceso:

Es importante conocer cómo detener un hilo de proceso para que en caso que una solicitud se haya vuelto demasiado extensa podamos detenerla sin perder la información recogida hasta ese momento.

Para hacerlo validamos que el hilo esté inicializado y realmente esté en ejecución, de este modo evitamos procesos y errores innecesarios. Posteriormente llamando su método de Abort() detemos el proceso que este hilo esté ejecutando. Este proceso se hace dentro de un try/catch para poder capturar cualquier excepción que ocurra en el proceso de forzar una detención.

try

{

if (\_thread != null && \_thread.IsAlive)

\_thread.Abort();

}

catch (ThreadAbortException ex)

{

MessageBox.Show(@"Se ha detenido el proceso. " + ex.Message);

}

*Se recomiendo el uso del método de clase “\_cells.SetCursorDefault ();” para indicar al usuario mediante la animación del cursor que se detuvieron los procesos en segundo plano.*

A continuación dejamos un ejemplo de este proceso que está sujeto a unos controles de tipo button. En el primero iniciamos el llamado al hilo y lo ejecutamos de una vez, y en el segundo la opción de detener el proceso.

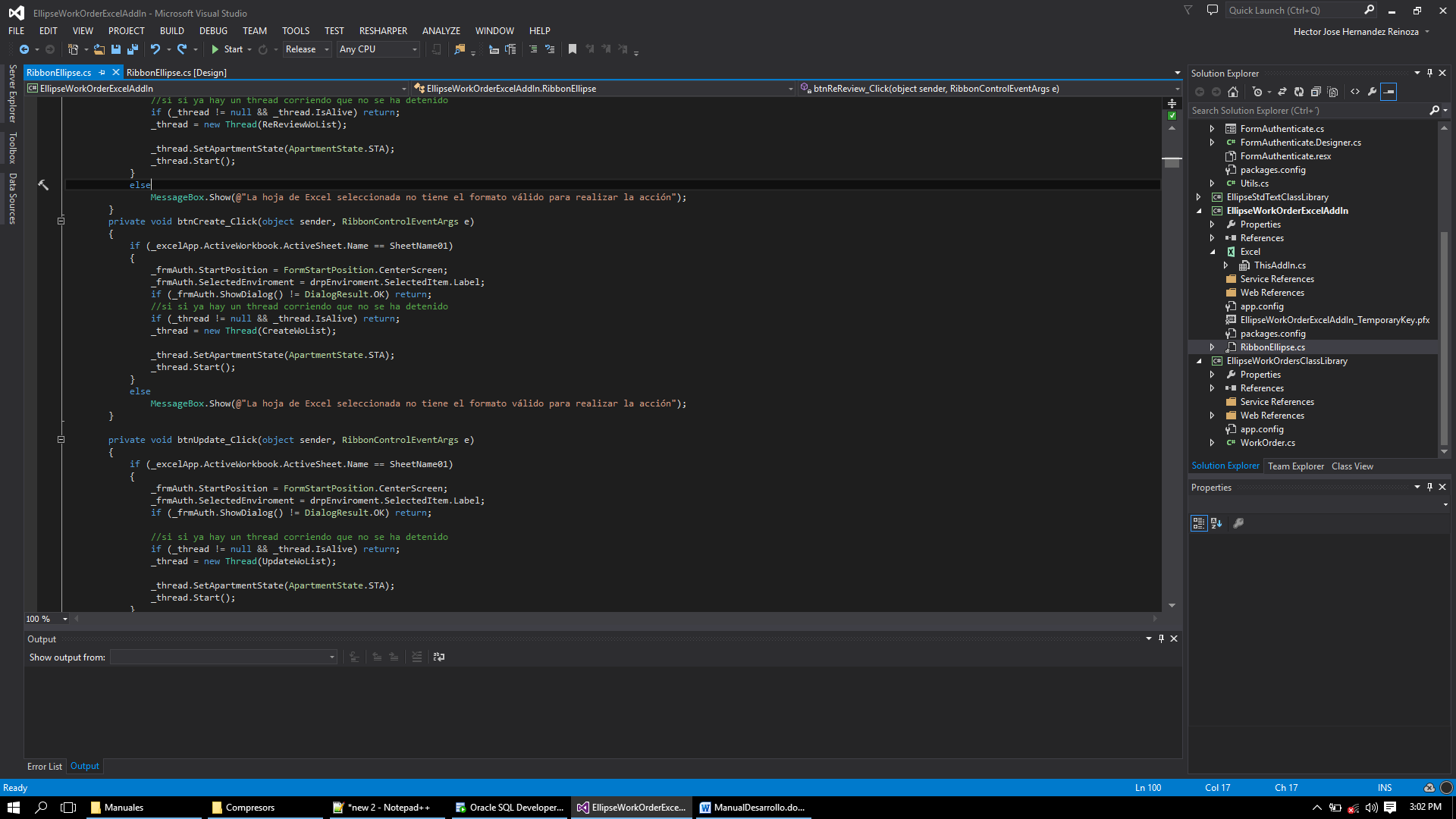


Imagen 32. Inicialización y ejecución de un método en un hilo de proceso

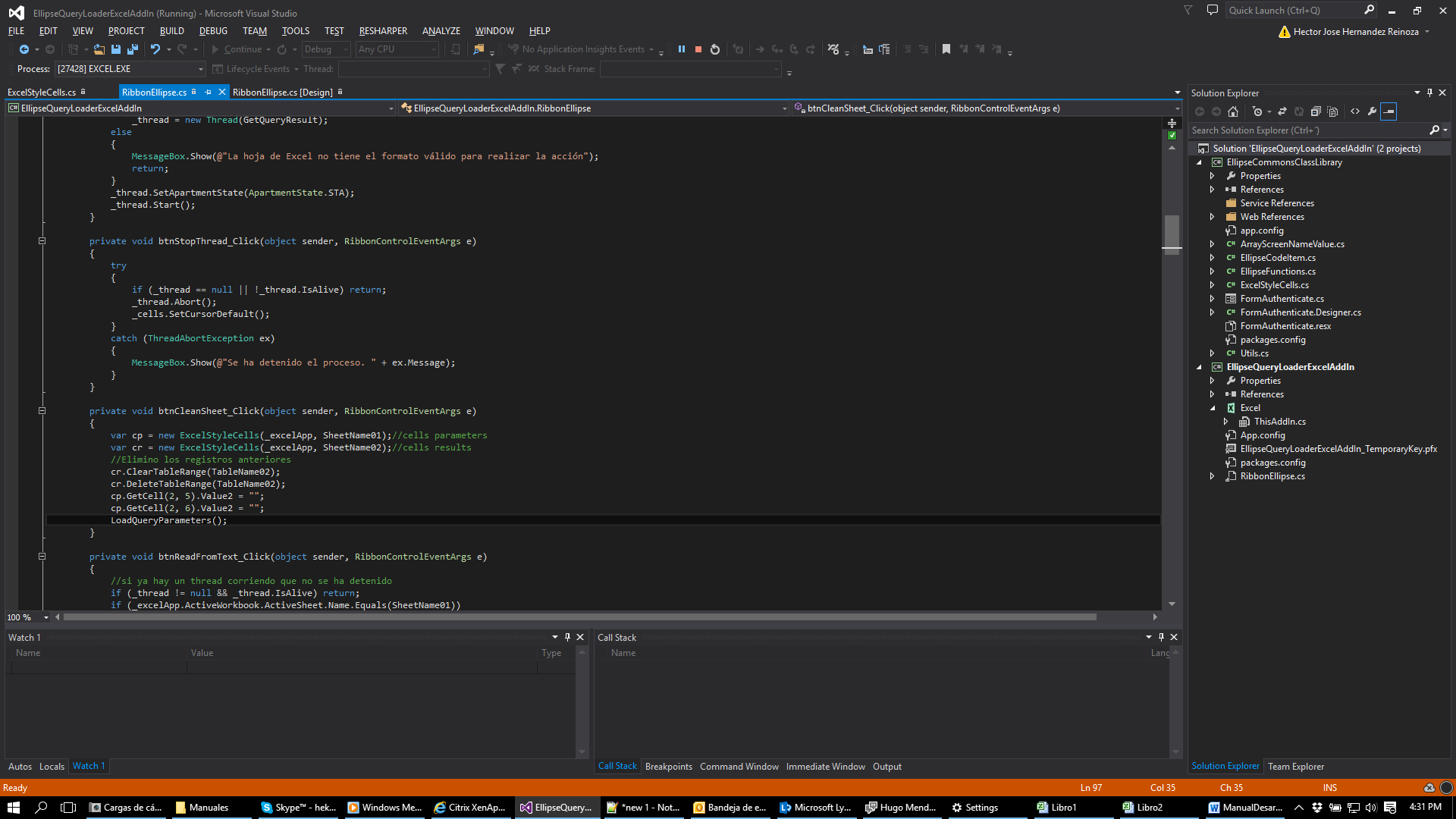


Imagen 33. Detención forzosa de un hilo de proceso

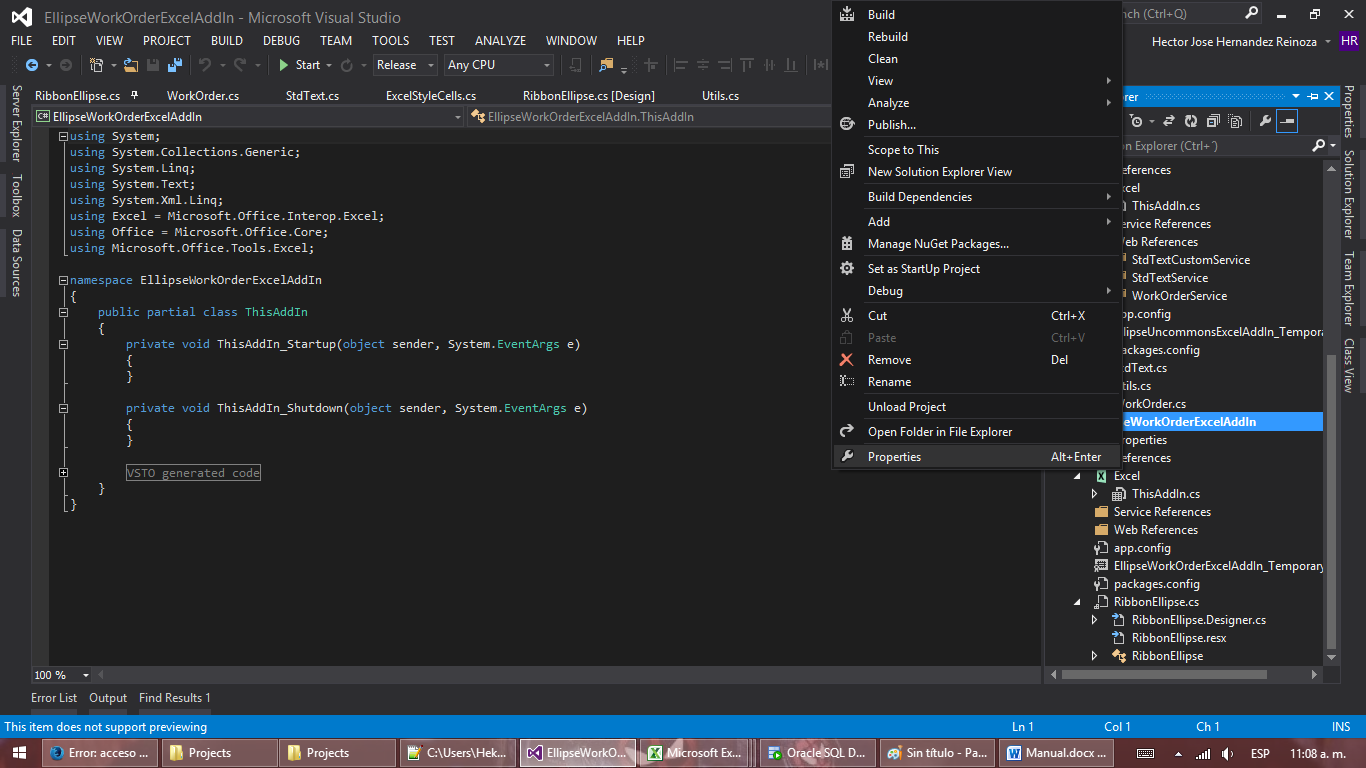
*Para efectos de desarrollo de los loaders en Excel, hay que tener en cuenta que si el hilo está consultando o escribiendo información en una hoja de un libro lo haga haciendo uso de una referencia absoluta de la clase ExcelStyleCell. Esto con el fin de evitar que si el proceso está en ejecución y el usuario cambia de hoja o de libro, interfiera con el proceso del hilo al cambiar la hoja activa (ver 7.6).*

## CONTROL DE VERSIONES Y PUBLICACIÓN DEL PROYECTO

### Control de versiones

Es importante la correcta gestión de versiones para que siempre se mantenga actualizado el AddIn ante cualquier cambio que se realice.

Para esto desplegamos el menú de nuestro proyecto [1] le damos *Properties [2]*

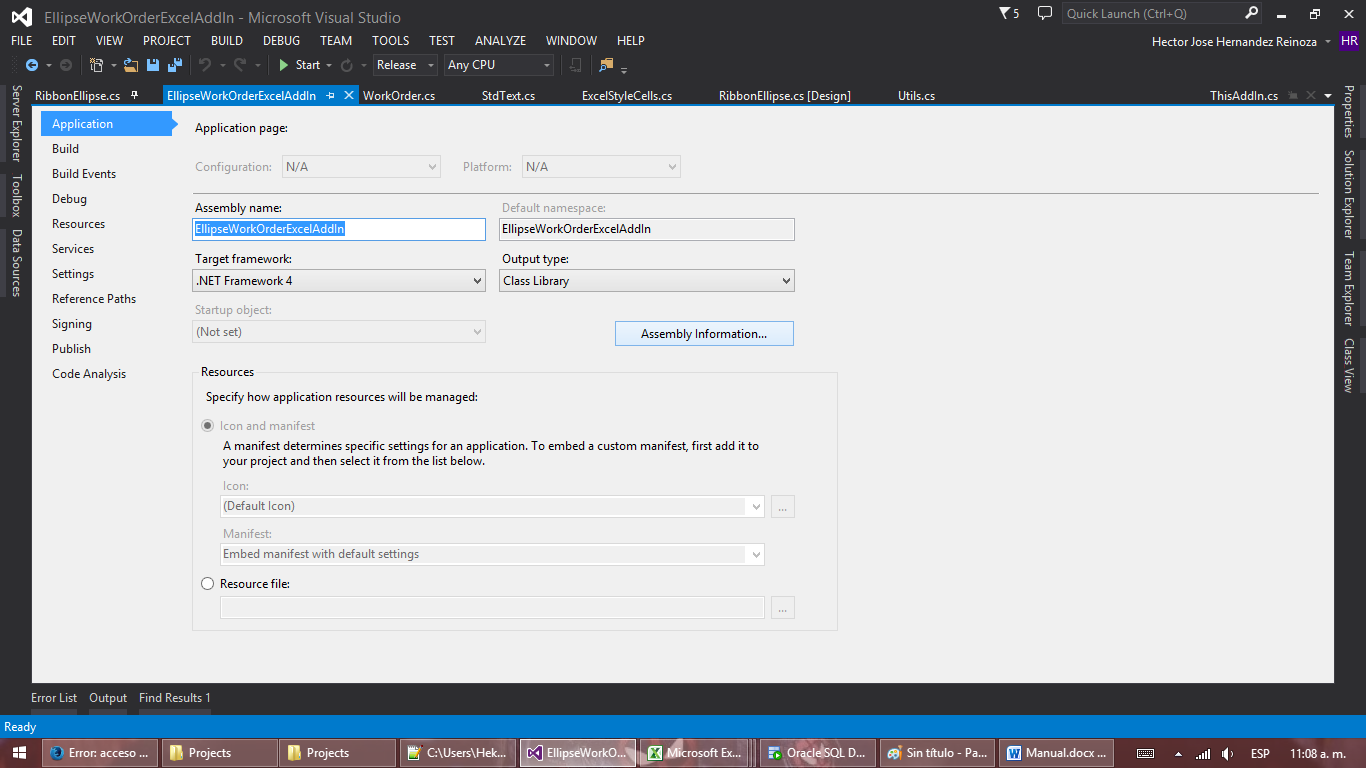


**1**

**2**

Imagen 34 Despliegue de propiedades del proyecto

Seleecionamos la opción de *Application [3]* en la barra lateral izquierda y hacemos clic en el botón *de Assembly Information [4].*

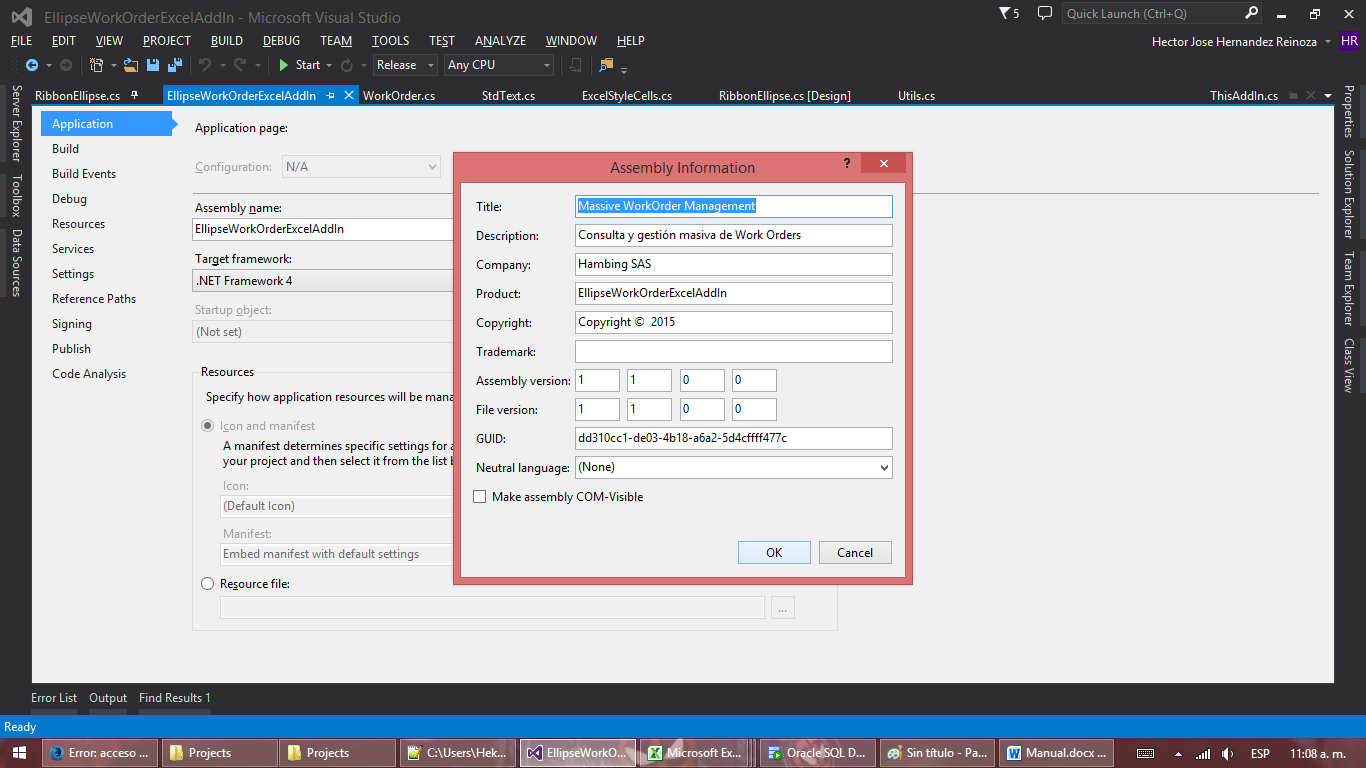


**3**

**4**

Imagen 35. Propiedades del proyecto

Gestionamos la información según deseamos teniendo en principal cuenta el número de la versión [5] y seleccionamos el botón OK [6].

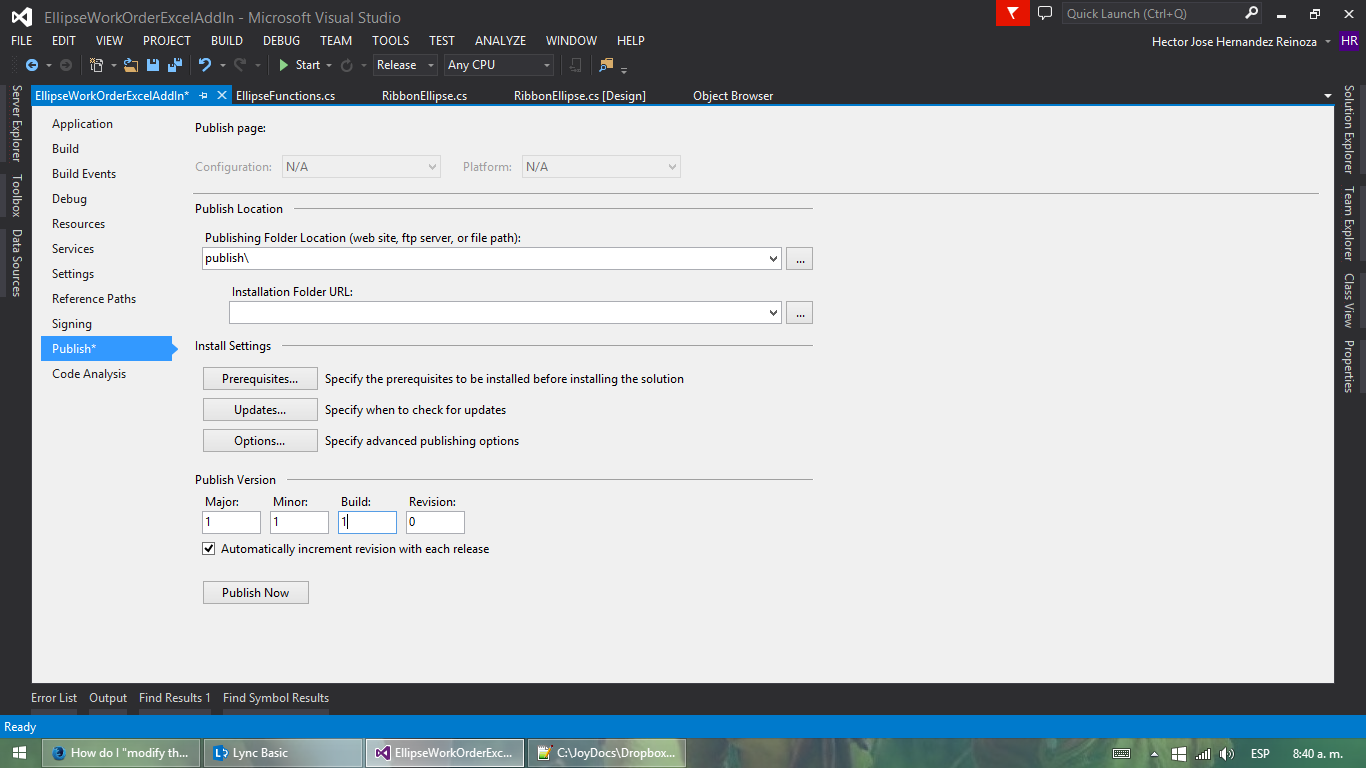


**5**

**6**

Imagen 36. Información de Ensamblado

Al mismo tiempo, seleccionamos ahora la opción de *Publish* [7] y ponemos el mismo número de versión en la sección de *Publish Version [8]*



**7**

**8**

Imagen 37. Información de Publicación

Finalmente podremos ingresar en la ruta del proyecto y en la carpeta publish encontraremos los ficheros de publicación del AddIn



Imagen 38. Archivos de Publicación