ECOSIMPRO: ENTORNO VIRTUAL SOBRE EL QUE DESARROLLAR HERRAMIENTAS SCADA PARA PROCESOS QUÍMICOS

Luis Taboada Antelo, Marcos Villafín Martínez, Julio Rodríguez Banga y Antonio Álvarez Alonso
Grupo de Ingeniería de Procesos, **IIM-CSIC**
C/Eduardo Cabello 6, 36208- Vigo (España)

Resumen

En este trabajo se pretenden mostrar las posibilidades que hacen de ECOSIM una base adecuada para el desarrollo de aplicaciones SCADA, las cuales se pueden emplear, posteriormente, en diversos procesos químicos.

Palabras Clave: EcosimPro, SCADA, Modelo, Aplicación, Servidor, Cliente.

1 INTRODUCCIÓN

La mayor parte de las plantas industriales tienen un Centro de Control donde se encuentra instalada una aplicación SCADA (*Supervisory Control And Data Adquisition*) que, como su nombre indica, se encarga de la adquisición de datos para la supervisión y control de los procesos industriales.

Actualmente existe una gran oferta de *software* de adquisición, supervisión y control. La evolución de este *software* sigue una triple tendencia:

- Intercomunicación entre aplicaciones.
- Estandarización de las comunicaciones con los dispositivos de campo.
- Adopción del entorno de comunicación Internet.

Entre los elementos que caracterizan la oferta actual de software de supervisión y control destaca la programación en *Visual Basic para Aplicaciones* (VBA) que, junto con la tecnología *ActiveX*, proporciona una manera práctica, rápida y eficaz de desarrollar un entorno personalizado. Incorporar un control *ActiveX* en una pantalla supone añadir un objeto con código asociado que realiza una determinada función de forma totalmente integrada dentro de la aplicación que estamos tratando.

En este trabajo se realizan 2 tipos de aplicaciones, según se sitúen en el *servidor* (proveedor de datos) o en un equipo remoto denominado *cliente* (receptor de datos):

a. La aplicación SCADA, emplazada en el *servidor* de planta, ha sido desarrollada sobre la base de un modelo y una simulación en ECOSIMPRO.

b. Con respecto a la aplicación *cliente*, que permite la monitorización y control remoto, se debe tener en cuenta que debe cumplir una serie de requisitos:

   - La interfaz gráfica debe ser lo más parecida a la interfaz gráfica de la aplicación SCADA.
   - Debe permitir visualizar el estado del sistema SCADA así como poder interactuar con él mediante el envío de directivas de control.
   - Y, finalmente, debe de ser segura y eficaz en la mayor medida posible.

Se crea una base de datos que actúa de intermediaria entre la aplicación SCADA y la aplicación cliente. En esta base de datos se irán registrando todos los valores de los parámetros de la aplicación SCADA y, de esta forma, la aplicación remota podrá obtener dichos valores en tiempo real, así como enviar directivas de control que la aplicación servidor monitoriza, leyendo en la base de datos y corrigiendo las posibles variaciones operacionales.

Figura 1: Esquema general del sistema

Todo este entorno se genera gracias a las facilidades de conectividad con otras aplicaciones que presenta ECOSIMPRO, ya que incorpora un *ActiveX* (EcoViewer.dll) que proporciona al programador una serie de herramientas que hacen posible la creación

de cualquier *interface* donde se muestren los resultados de los experimentos implementados en ECOSIMPRO, con la posibilidad de poder manipular las variables que intervienen en los mismos.

2 MODELADO EN ECOSIMPRO

ECOSIMPRO, como herramienta matemática de modelado y simulación de sistemas dinámicos, permite el estudio de procesos reales, tanto a pequeña como a gran escala. Tomando este punto de partida, resulta evidente que es fundamental el disponer de un modelo matemático lo más adecuado a la realidad como sea posible. De esta forma, podrá reproducirse fielmente cualquier planta mediante el lenguaje EL de ECOSIMPRO.

Los pasos a seguir serán:

1. Implementación del modelo matemático en ECOSIMPRO.

2. Generación de los experimentos de acuerdo con las necesidades de la aplicación SCADA a desarrollar, teniendo en cuenta tanto los valores iniciales como los límites de las variables implicadas.

La principal necesidad en el modelo radica en la sincronización del tiempo de simulación y el real, porque se plantea como objetivo de este trabajo el hecho de que la aplicación SCADA desempeñe su función en tiempo real. Para ello, a la hora de configurar los experimentos, en la sección BODY, debe incluirse la siguiente sentencia

```
BODY
FOLLOW_RT= TRUE
...
END EXPERIMENT
```

Esto facilita la interactividad entre las aplicaciones servidor y cliente, puesto que si el proceso de cálculo se realiza en tiempo computacional, al decidir ejercer cualquier acción de regulación, ésta sería muy difícil de ubicar en un momento temporal específico para dicha acción.

Asimismo, es fundamental el proceso de inicialización de las variables principales (controladas, manipuladas y las asociadas a perturbaciones) empleadas en el modelo, puesto que serán el punto de partida para la generación de la aplicación SCADA.

2.1 SISTEMA DE DOS TANQUES EN SERIE CON RECIRCULACIÓN

A fin de ejemplificar lo expuesto anteriormente, se desarrolla un sistema sencillo consistente en dos tanques de mezcla completa conectados en serie y con una recirculación del tanque 2 al tanque 1. En ambos tanques se pretende establecer un control del nivel (mediante el control de las alturas h_1 y h_2) y un control de la temperatura de salida de los tanques (T_{1s} y T_{2s}), la cual coincide con la temperatura en cualquier punto al considerarse mezcla completa. Este tipo de control es representativo del control de la calidad del producto en muchos procesos químicos.

Un esquema del problema se puede ver en la siguiente figura:

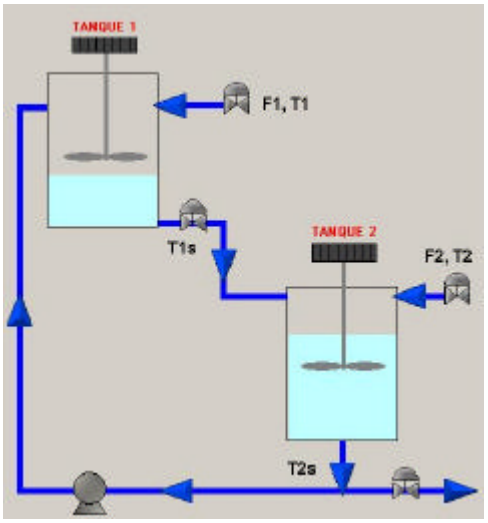

Figura 2.- Esquema del sistema de tanques en serie

En este sistema se consideran 4 *variables controladas* (h_1, h_2, T_{1s}, T_{2s}) y se dispone de 5 *variables manipuladas*, representadas por los grados de apertura de las diferentes válvulas y la bomba de reflujo representadas en la Figura 2.

Como posibles perturbaciones que puedan afectar a las variables controladas, se van a considerar únicamente las temperaturas de las alimentaciones frescas de ambos tanques (T_1 y T_2) y posibles cambios en los puntos de consigna de dichas variables a controlar (h_{1r}, h_{2r}, T_{1sr} y T_{2sr}).

Para la selección de los lazos de control a implementar en este sistema multivariable, se realizó un estudio mediante el **RGA** (***Relative Gain Array***) o matriz de ganancias relativas, definido en [1].

El RGA se define como **una matriz de números**. El elemento *ij* en la matriz se llama β_{ij}. Es la relación de la ganancia en estado estacionario entre la *i-ésima* variable controlada y la *j-ésima* variable manipulada, cuando todas las demás variables manipuladas son constantes, dividida por la ganancia en estado estacionario entre las dos mismas variables cuando todas las otras variables controladas son constantes:

$$\beta_{ij} = \frac{\left[\partial x_i / \partial m_j\right]_{m_k}}{\left[\partial x_i / \partial m_j\right]_{x_k}} \qquad (1)$$

Los elementos del RGA pueden calcularse, para cualquier sistema, usando la siguiente ecuación:

β_{ij} = *(elemento ij-ésimo de* $\underline{\underline{K_p}}$ *)(elemento ij-ésimo de* $\left[\underline{\underline{K_p^{-1}}}\right]^T$ *)* (2)

siendo K_p la matriz de ganancias en estado estacionario en este sistema.

Cuanto más cerca de 1 estén los valores de β_{ij} , menor interacción habrá (menos diferencia entre el sistema en lazo abierto y cerrado), de manera que los pares de variables se escogen en función de cuales tengan los elementos en el RGA más próximos a 1.

Para el proceso sometido a estudio, los lazos de control obtenidos son:

Tabla 1: Pares de variables seleccionados

Número de lazo	**Variable controlada**	**Variable manipulada**
Lazo 1	h_1	Apertura de la válvula de salida del tanque 1
Lazo 2	h_2	Apertura de la válvula de alimentación fresca al tanque 2 (F_2)
Lazo 3	T_{1s}	Apertura de la válvula de alimentación fresca al tanque 1 (F_1)
Lazo 4	T_{2s}	Bombeo de corriente de recirculación

Como resultado de esta selección, la configuración del proceso queda como se muestra en la figura 3:

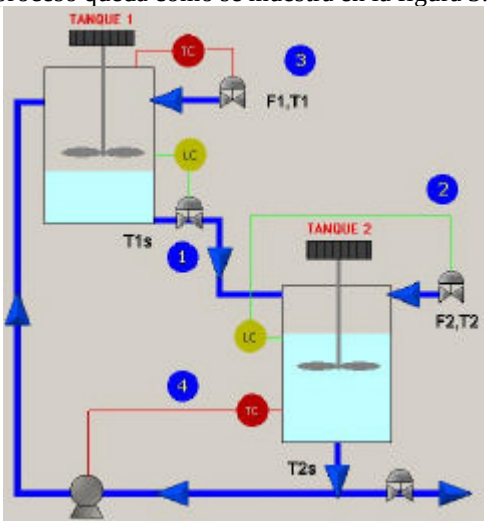

Figura 3: Esquema del sistema considerando lazos de control

Todos los controladores que se utilizan se consideran del tipo PI (Proporcional-Integral), siendo ajustados por el método *de respuesta de frecuencia*. Para más información sobre las diferentes alternativas existentes para determinar los parámetros de diferentes tipos de controladores, ver [5].

Una vez desarrollado todo el modelo matemático y seleccionados lazos y parámetros de control, el siguiente paso es la implementación del mismo en ECOSIMPRO

Para ello, se crea un componente, denominado *tanques_en_serie*, en el cual se reflejan los balances de materia y energía en ambos tanques junto con las consignas de control, de manera que el proceso queda fielmente reflejado. El código completo de este sencillo componente se puede consultar en el APÉNDICE A.

Mediante una serie de puertos ya predefinidos en los controladores que incorpora la librería CONTROL de ECOSIMPRO, se interconectan los cuatro controladores con el componente *tanques_en_serie*, generando un nuevo componente, denominado *control_tanques _en_serie*, sobre el que se desarrollen los experimentos que servirán de base a la aplicación SCADA.

El código para este componente se muestra a continuación:

```
-----------------------------------------------------------------
USE CONTROL

COMPONENT control_tanques_en_serie

TOPOLOGY
        tanques_en_serie TS
        Cntrl_pi PI (k=-0.175, Ti=12.64, u_min=-1)
        Cntrl_pi PI2(k=-0.109, Ti=11.375, u_min=-1)
        Cntrl_pi PI3(k=-0.10, Ti=10.554, u_min=-1)
        Cntrl_pi PI4(k=-0.12, Ti=10.04, u_min=-1)

        CONNECT TS.nivel1 TO PI.s_var
        CONNECT TS.href1 TO PI.s_set
        CONNECT PI.s_out TO TS.he
        CONNECT TS.nivel2 TO PI2.s_var
        CONNECT TS.href2 TO PI2.s_set
        CONNECT PI2.s_out TO TS.F2e
        CONNECT TS.T1sal TO PI3.s_var
        CONNECT TS.T1sref TO PI3.s_set
        CONNECT PI3.s_out TO TS.F1e
        CONNECT TS.T2sal TO PI4.s_var
        CONNECT TS.T2sref TO PI4.s_set
        CONNECT PI4.s_out TO TS.be

END COMPONENT
-----------------------------------------------------------------
```

Los valores de las ganancias y los tiempos integrales definidos en el componente serán los empleados por defecto por la aplicación.

El experimento base para el desarrollo de la aplicación SCADA es el siguiente:

```
-----------------------------------------------------------------
 EXPERIMENT exp2 ON control_tanques_serie.default

DECLS

INIT    -- set initial values for variables
    -- Dynamic variables
  TS.h1 = 1.361
  TS.h2 = 2.777
  TS.x = 110.747
  TS.r = 42.682
  PI.vi = 0
  PI2.vi = 0
  PI3.vi = 0
  PI4.vi = 0

BOUNDS   -- set expressions for boundary variables: v =
          f(t,...)
  TS.T1 = 0
  TS.T1sr1 = 0
  TS.T2 = 0
  TS.T2sr1 = 0
  TS.hr1v = 0
  TS.hr2v = 0

BODY
  FOLLOW_RT= TRUE
  TIME = 0
  TSTOP = 360000
  CINT = 0.5
  INTEG()

END EXPERIMENT
-----------------------------------------------------------------
```

Como principales características de este experimento destacan:

1. La ya citada sentencia de correspondencia entre el tiempo de simulación y el tiempo real.
2. La ausencia de la orden de generación de un informe de resultados, puesto que no es necesario. Los datos se irán guardando automáticamente en la base de datos que servirá de centro del sistema SCADA.
3. La duración de la simulación se considera suficiente a efectos prácticos.
4. El tiempo de muestreo se elige pequeño ante la rapidez con el que el sistema responde ante las perturbaciones, lo que haría que para tiempos mayores, no se observara toda la respuesta dinámica del mismo, perdiendo calidad la monitorización.

Al compilar dicho experimento, se genera un archivo ejecutable (exp2.exe), el cual realizará la función de la planta de proceso considerada mediante el envío y/o lectura de los valores de las variables que intervienen en la simulación.

A partir de este ejecutable y con las herramientas disponibles en ECOSIMPRO, resulta de fácil desarrollo el sistema de monitorización, control y supervisión necesario para el proceso considerado.

3 DISEÑO Y DESARROLLO DE LA APLICACIÓN SCADA

Visual Basic para Aplicaciones es el lenguaje de programación incorporado por las aplicaciones de *Microsoft Office* y ofrece diversas ventajas:

- Está muy extendido.
- Es aceptado por diversos fabricantes, por lo que se va convirtiendo en un estándar de hecho.
- Presenta una muy buena relación entre potencia y dificultad de aprendizaje y uso.

Para llevar a cabo el proceso de desarrollo, se empleará el entorno de programación *Microsoft Visual Basic 6.0*.

3.1 COMUNICACIÓN ENTRE ECOSIMPRO Y VISUAL BASIC

Como ya se mostró en la introducción, la tecnología *ActiveX* proporciona una manera sencilla de poder generar una *interface* en la cual se ejecuten los experimentos de ECOSIMPRO, facilitando las herramientas necesarias para que, desde el entorno de programación, se pueda interactuar con dichos experimentos.

En este caso particular, dicho *ActiveX* viene suministrado como una DLL (*Dynamic Link Library*) denominada EcoViewer.dll. Esta librería proporciona los objetos necesarios para llevar a cabo la programación de la aplicación en base al experimento generado a partir de un modelo implementado en ECOSIMPRO.

Habitualmente, la utilización del EcoViewer se realiza una vez que se ha instalado ECOSIMPRO, porque se necesitan algunas librerías dinámicas y otras que utiliza el Ecoviewer. Las simulaciones son de modelos realizados en ECOSIMPRO de tal forma que será necesario el ejecutable del modelo y los ficheros que necesita dicho ejecutable *(.txt)* y que directamente son obtenidos de la compilación del experimento. Este archivo txt es fundamental pues recoge todas las variables y ecuaciones que definen el modelo del proceso.

En este trabajo también se ha contemplado la posibilidad de que la aplicación no tenga dependencia directa con ECOSIMPRO, es decir, que en la máquina no esté instalado el software. Para solucionar este tema, se suministra un *setup.exe* facilitado por EA International, que se debe ejecutar para que se instale en dicha máquina todo lo necesario para que los entornos desarrollados en VB a partir de modelos de ECOSIMPRO funcionen correctamente.

3.2 DISEÑO DE LA INTERFACE GRÁFICA DE LA APLICACIÓN SCADA

El diseño de la interface gráfica de la aplicación SCADA está basada en el diseño real del proceso en el que se integra así como en las necesidades reales de supervisión y control de determinadas variables.

En el ejemplo de los tanques en serie, se ha optado por un gráfico dinámico generado en *Visual Basic 6.0*, tanto para el servidor como para el cliente.

La aplicación *servidor* se muestra en la figura 6. En ella también se indican las principales acciones que el usuario puede ejerce sobre el entorno gráfico.

Dicha aplicación presenta dos modos de operación básicos, cada uno de los cuales se desarrolla sobre la base de un experimento previamente generado:

a. **Modo sin control:** Al seleccionar la opción *Sin control* de las *Opciones de Control* de la ventana gráfica, el usuario puede estudiar la respuesta dinámica del sistema frente a perturbaciones tanto en los caudales y las temperaturas de las alimentaciones frescas de los tanques, observando la rapidez con la que el sistema manifiesta y propaga estas perturbaciones. Es un paso fundamental para la posterior implantación de sistemas de control.

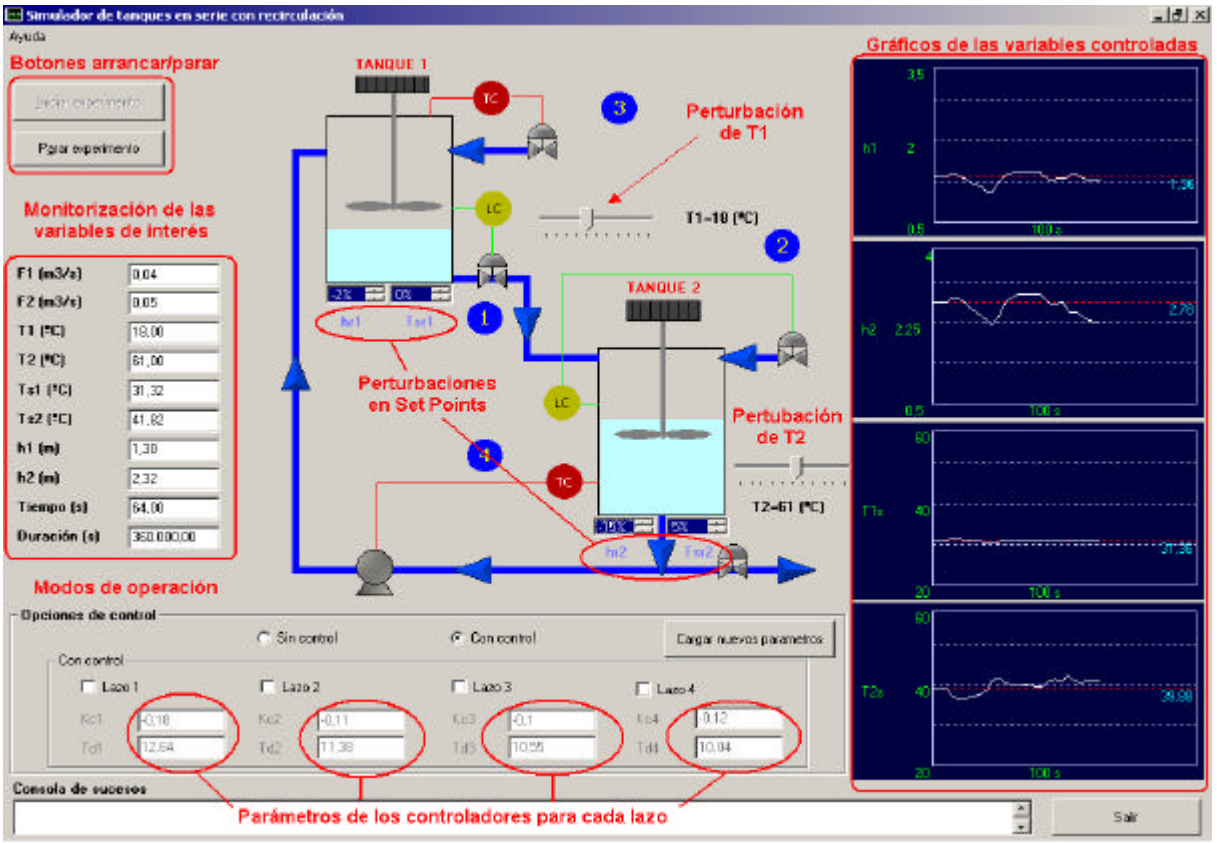

Figura 6: Ventana gráfica de la aplicación servidor

b. **Modo con control:** Si se escoge el modo *Con Control* de las *Opciones de Control*, la ventana gráfica cambia, puesto que se muestran los diferentes lazos de control implementados. Asimismo, permite la edición y modificación de los parámetros de ajuste de cada uno de los controladores.

En cuanto a las perturbaciones, ahora ya no resulta posible el manipular los caudales de alimentación fresca, puesto que pasan a ser variables manipuladas de los lazos 2 y 3. Sin embargo, se ofrece la posibilidad de introducir modificaciones en los puntos de consigna de cada una de las variables controladas.

Para más información acerca de los modos de operación, el programa dispone de un menú de ayuda.

En ambos modos de operación, resulta de vital importancia la parte de monitorización que se ofrece en la interface, en forma de gráficas y tablas dinámicas que se actualizan con el tiempo, puesto que a partir de dicha información se tomarán las posibles decisiones de control a ejercer sobre el sistema.

En cuanto al diseño de la interface *remota* (basado en la aplicación servidor), se ha optado por simplificar el entorno gráfico eliminando el modo SIN CONTROL, puesto que este nivel interesa, fundamentalmente, la monitorización y control del proceso, no el estudio dinámico del mismo, el cual se supone que es un paso previo a la implantación de la aplicación SCADA.

3.3 FUNDAMENTOS DE PROGRAMACIÓN DE LA INTERFACE

Como punto de partida para la creación de la interface se debe, en primer lugar, hacer referencia a la librería EcoViewer.dll, que, como antes se indicó, es el componente ActiveX encargado de establecer la comunicación Visual Basic-ECOSIMPRO. Para llevar esto a cabo, dicha DLL debe estar registrada, operación ya realizada si ECOSIMPRO está instalado o si se ha instalado el *setup.exe* mencionado con anterioridad.

Partiendo de la premisa de que esta condición se cumple, y ya dentro del entorno de programación, se debe acceder al menú *Proyecto* > *Referencias* de VB. En el cuadro de diálogo *Referencias* es necesario seleccionar la opción *EcoViewer Monitor ECOSIMPRO 3.2.*, como se muestra en la figura 7. Se acepta y, en estos momentos, el programador ya se encuentra en disposición de emplear este objeto.

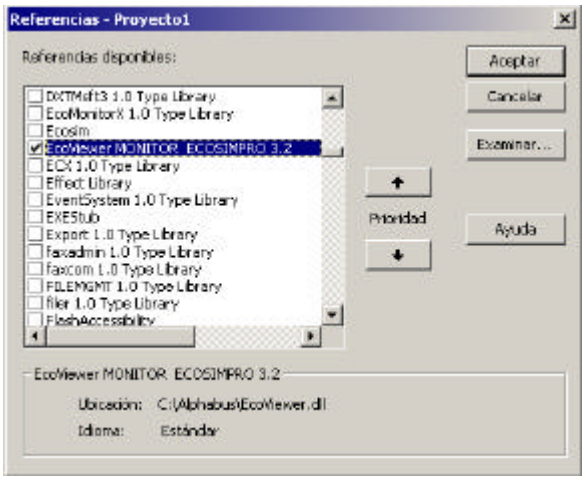

Figura 7: Cuadro de diálogo para la selección de la DLL

A continuación, se realiza una breve descripción de cómo cargar en una variable objeto un experimento realizado con ECOSIMPRO. Para explicar dicho procedimiento, se irá ejemplificando con el sistema sencillo de tanques en serie:

1º. *Declaración de la variable objeto* (Experimento):

```
Public WithEvents Experimento As EcoViewer.EcoSimViewer
```

2º. *Creación de una instancia de la variable objeto* Experimento *(en el* Form_Load, evento que se desencadena al cargar el formulario):

```
Set Experimento = New EcoViewer.EcoSimViewer
```

3º. *Asignación de un cuadro de texto* (Textbox) *para el seguimiento de los experimentos:*

```
Set Experimento.WindowOuput = Text1
```

4º. *Inicialización del experimento:*

```
ArchivoExp = App.Path & "\exp" & Expe & ".exe"
Experimento.LaunchSimulationFile ExpeFile
```

Mediante la primera sentencia, en la variable **ArchivoExp**, introducimos la ruta del ejecutable del experimento. *App.Path* devuelve la ruta de la carpeta donde se ubica la aplicación, de forma que se debe situar el experimento en dicha carpeta.

La variable auxiliar **Expe** se utiliza para diferenciar el ejecutable empleado en los dos modos de operación considerados, de manera que si se elige la opción SIN CONTROL, **Expe** será igual a 1 (cargando el archivo exp1.exe) y en el caso CON CONTROL, **Expe** será igual a 2 (exp2.exe).

La segunda sentencia se encarga de enlazar el experimento con la variable objeto.

Si el experimento se ha cargado de forma satisfactoria, se avanza a la siguiente etapa.

5º. *El evento* Experimento_Connected:

```
v.Vars.Add Vars.Item(32).Name
LoadValores v
End If

If OptControl(2).Value = True Then
v.Vars.Add Vars.Item(61).Name
v.Vars.Add Vars.Item(64).Name
v.Vars.Add Vars.Item(69).Name
v.Vars.Add Vars.Item(73).Name
v.Vars.Add Vars.Item(37).Name
v.Vars.Add Vars.Item(52).Name
v.Vars.Add Vars.Item(82).Name
v.Vars.Add Vars.Item(84).Name
v.Vars.Add Vars.Item(132).Name
```

```
v.Vars.Add Vars.Item(133).Name
.Vars.Add Vars.Item(89).Name
v.Vars.Add Vars.Item(77).Name
v.Vars.Add Vars.Item(8).Name
v.Vars.Add Vars.Item(22).Name
v.Vars.Add Vars.Item(36).Name
v.Vars.Add Vars.Item(51).Name
LoadValores v
v

Set v = Nothing

Set v = Views.Add("CONTROLADORES")
 v.Vars.Add Vars.Item(7).Name
v.Vars.Add Vars.Item(1).Name
v.Vars.Add Vars.Item(21).Name
v.Vars.Add Vars.Item(15).Name
v.Vars.Add Vars.Item(35).Name
v.Vars.Add Vars.Item(29).Name
v.Vars.Add Vars.Item(50).Name
v.


Vars.Add Vars.Item(44).Name
LoadParametros v

End If

End Sub
```

Este evento se desencadena si la conexión experimento-objeto se ha realizado con éxito. De ser así, la variable **StateConnect** toma el valor 4, con lo cual se está en condiciones de generar las *views* (vistas), creando entonces variables de objeto que soporten el contenido de estas vistas (**Views**) así como de las variables asociadas a cada una de ellas (**Vars**).

En el caso particular del ejemplo de los tanques, al poder elegir entre dos modos de operación diferentes, debe añadirse una sentencia que haga distinción entre cada uno de ellos. Al disponer en la interface gráfica de un botón de selección de *Opciones de Control* (ver Figura 6), el activar uno u otro caso origina que el control de opción tome el valor verdadero. Por ejemplo, OptControl(1).Value = True será cierta si se selecciona la opción SIN CONTROL.

Según el sistema seleccionado, el programa opera asignando una view en la que se cargan los nombres de las variables de interés desde el punto de vista de la aplicación SCADA. Para el caso CON CONTROL es necesaria una segunda vista (CONTROLADORES), en la cual se cargan las variables correspondientes a los parámetros de los controladores. Esto facilitará el manejo de las variables, puesto que se separan las acciones de monitorización de las posibles actuaciones a nivel de controladores. La otra view (VISTA2) se crea teniendo en cuenta las variables que van a ser necesarias según el caso (controlado o sin controlar).

La instrucción que permite añadir el nombre de una variable a la view es

```
v.Vars.Add Vars.Item(n).Name
```

donde n es la posición que a dicha variable asigna ECOSIMPRO en el modelo. Dichas posiciones vienen reflejadas en el archivo de texto que precisa el ejecutable (con la extensión .default).

Tras añadir las variables deseadas a las views, se realiza una llamada a un procedimiento que asigna el valor de cada variable de la vista, pasada como parámetro, a una matriz numérica . El código asociado a esta operación, ejemplificando para la VISTA2, se muestra a continuación:

```
Private Sub LoadValores(v As ecmView)

Dim vVars As ecmViewVars
Dim vVar As ecmViewVar
Set vVars = v.Vars
Dim i As Integer
i = 1
    For Each vVar In vVars
      NombresVal(i) = vVar.Name
   ValoresIni(i) = vVar.ExpVar.Value
    i = i + 1
    Next

End Sub
```

Donde ValoresIni(i) es la matriz numérica que almacena los valores de las variables deseadas y NombresVal(i) es una matriz donde se guardan los nombres de las variables seleccionadas, la cual se utilizará en la asignación de valores.

Ciertas variables precisan ser inicializadas para establecer un punto de partida en el desarrollo de la simulación. Esta tarea puede realizarse bien en el experimento o mediante código en el programa. Esta segunda alternativa presenta la ventaja de que se pueden modificar las condiciones iniciales sin tener que generar nuevamente el experimento, tarea imposible de realizar en una máquina en la cual no esté instalado ECOSIMPRO. La siguiente función realiza la tarea de asignar nuevos valores, de manera que para inicializar ciertas variables basta con llamar a la función pasando como parámetro (i) la posición que ocupa dicha variable en la matriz de valores (ValoresIni). Para llevar a cabo dicha asignación es preciso conocer el nombre de las variables a modificar. Este es el motivo por el cual se creó una matriz que almacene los nombres.

```
Dim NuevoValorVariable As Variant
Dim mVarType As Variant

Dim v As ecmExpVar
```

```
On Error GoTo MiError
NuevoValorVariable = ValoresIni(i)
Set v = Nothing

mVarNombre = NombresVal(i)
AsignarValor = 0
If mVarNombre <> vbNullString Then
Set v    =Experimento.ecmExpVars
(mVarNombre)
```

```
   mVarType = v.TypeDef
Select Case mVarType
      Case ecmString
AsignarValor = CStr(NuevoValorVariable)
      Case ecmReal
AsignarValor = CDbl(NuevoValorVariable)
      Case ecmInteger
AsignarValor = CLng(NuevoValorVariable)
      Case ecmBoolean
AsignarValor = CBool(NuevoValorVariable)
      Case Else
AsignarValor = 0
   End Select
      End If
      v.Value = AsignarValor
      On Error GoTo 0
      Exit Function

    MiError:
      If v Is Nothing Then
        MsgBox "La  variable no pertenece al
experimento", vbInformation, "ALERTA"
      End If
      AsignarValor = 0
      On Error GoTo 0

    End Function
```

El desarrollo de esta función es simple: se almacenan el valor y el nombre de la variable tratada en dos variables temporales, de manera que mediante su nombre se pueda determinar el tipo de dato a través de una función de ECOSIMPRO (*TypeDef*) y, a partir de aquí, asignar un valor conforme a ese tipo.

6º. *Arrancada, parada y pausa del experimento:*

Alcanzado este punto en el desarrollo de la aplicación, se está en condiciones de iniciar el experimento. La instrucción que lo permite es la siguiente:

Lo visto para la **VISTA2** sería también aplicable a las demás vistas generadas, bien en el ejemplo presentado **(CONTROLADORES)** bien en otros sistemas estudiados.

```
Experimento.Run
```

Esta misma sentencia reanuda el experimento en caso de que éste se hubiera pausado previamente.

De igual forma, es posible pausar y/o parar la simulación de manera sencilla gracias a las siguientes sentencias

```
Experimento.Break
Experimento.EndSimulate
```

Una manera muy simple de interactuar con estas sentencias es asociarla a botones, como es el caso del ejemplo mostrado en este trabajo. En el mismo, todos los deslizadores y ventanas de selección de perturbaciones (ver Figura 6) presentan un patrón común de funcionamiento basado en la secuencia:

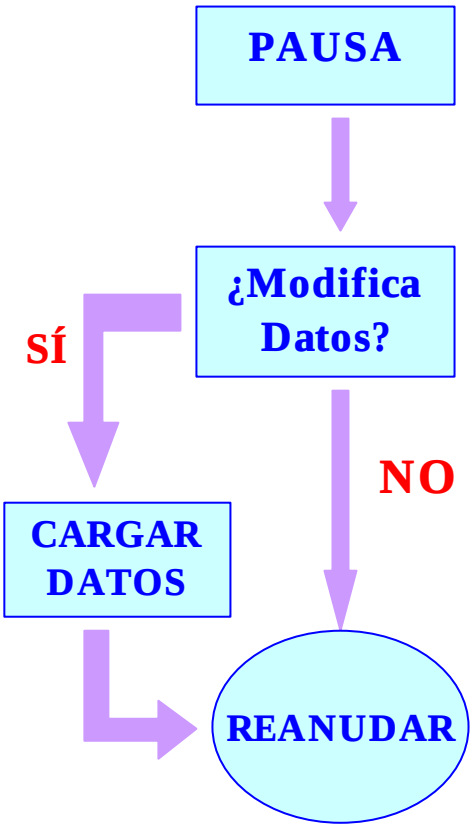

Figura 8: Secuencia lógica del funcionamiento de los controles del entorno gráfico

El aspecto novedoso aquí es el proceso de **carga de nuevos datos** al experimento. Es una operación sencilla puesto que es muy similar al proceso de inicialización de variables previamente descrito. La carga se realiza mediante la asignación del nuevo valor introducido por el usuario en el entorno gráfico (a través de deslizadores, ventanas de selección, etc) a la posición correspondiente de la matriz de valores previamente definida. Como el valor ha sido actualizado en la matriz, se toma la posición de dicho valor y se realiza una llamada a la función de

asignación, la cual introduce en el modelo el nuevo valor.

3.4 CONECTIVIDAD A TRAVÉS DE LA BASE DE DATOS

Una vez llegado hasta aquí, parece lógico el plantearse la posibilidad de la monitorización, supervisión y control remota del proceso. Esta necesidad es la causa que origina el desarrollo de la aplicación cliente.

El funcionamiento de la misma debe ser totalmente independiente de ECOSIMPRO, pues su origen de datos se halla en la aplicación servidor. Es por ello que se hace necesario el uso de una base de datos, que desempeñe las funciones de intermediaria entre el servidor y el cliente.

Para crear la base de datos se opta por emplear Microsoft Access 2000. Se ha optado por la creación de dos tablas, cada una de las cuales registrará de manera continua los valores de las variables pertenecientes a cada una de las vistas anteriormente definidas, denominadas respectivamente VISTA2 y CONTROLADORES.

TABLA **Parametros**

Nombre del campo	Tipo de datos
Kc1	Numérico
Td1	Numérico
Kc2	Numérico
Td2	Numérico
Kc3	Numérico
Td3	Numérico
Kc4	Numérico
Td4	Numérico

TABLA **ValoresIni**

Nombre del campo	Tipo de datos
F1	Numérico
F2	Numérico
T1	Numérico
T2	Numérico
T1s	Numérico
T2s	Numérico
h1	Numérico
h2	Numérico
Tiempo	Numérico
Duracion	Numérico
b	Numérico
k1	Numérico
h1r	Numérico
h2r	Numérico
T1sr	Numérico
T2sr	Numérico

Figura 9: Tablas definidas en la base de datos

En estas tablas, los valores se irán almacenando desde el servidor a partir de los valores generados por el experimento. El cliente accederá mediante una consulta a estos valores y los empleará para monitorizar el proceso. Asimismo, cuando una directiva de control sea implementada en el cliente, inmediatamente se verá reflejada en la base de datos mediante la actualización de los campos implicados y, por tanto, el servidor también está obligado a monitorizar la base de datos para detectar cambios en cualquier instante.

3.4.1 Interacción servidor-base de datos

Desde el punto de vista del servidor, éste precisa tomar una referencia para guardar los resultados de la simulación. Es por ello que se debe tener en cuenta cada cuanto tiempo se refrescan los valores.

El evento Experimento_RefreshView se desencadena cada vez que transcurre el tiempo de integración definido previamente en el cuerpo del experimento en ECOSIMPRO (CINT), actualizando los valores de las variables. Aprovechando esta circunstancia, es fácil entender que éste es el momento adecuado para comprobar si se ha actualizado el último registro en cada una de las tablas de la base de datos. Esto se consigue comparando campo a campo los dos últimos registros. Si estos son distintos, lógicamente, se habrá producido un cambio que habrá que tener en cuenta, asignando los valores del último registro.

El código general de este procedimiento es el que se muestra a continuación:

```
Private Sub Experimento_RefreshView(sViewName
As String, dTime As Double, VarValues As Variant)
 …
 AdoRs.MoveLast
 ValorX= AdoRs.Fields(i)
 AdoRs.MovePrevious
If AdoRs.Fields(i).Value <> ValorX Then
  ValoresIni(i) = ValorX
  AsignarValor (i)
End If
…
```

Donde se realizan las siguientes tareas:

1. Desplazamiento al último registro de la base de datos.

2. Asignación del valor del campo **i** a la variable **ValorX** (variable temporal).

3. Desplazamiento al penúltimo registro de la base de datos.

4. Comparación del valor del campo **i** en el penúltimo registro con la variable temporal. Si *son distintos* (se ha producido un cambio), lo asigna a la posición **i** de la matriz de valores de las variables seleccionadas y llama a la función de asignación.

De acuerdo con esos nuevos valores , se procede a la modificación de los dispositivos de lectura/escritura en la ventana gráfica (ventanas de monitorización, deslizadores, etc.).

El proceso de escritura en la base de datos también tiene lugar cuando se desencadena este evento. A continuación se detalla brevemente como realizar esta operación en Visual Basic:

```
Private Sub Experimento_RefreshView(sViewName
As String, dTime As Double, VarValues As Variant)
 …

 AdoRs.MoveLast
 AdoRs.MoveNext

          AdoRs.AddNew
          AdoRs.Fields(i) = VarValues(i)
          AdoRs.Update
...
```

Con esta serie de comandos tienen lugar las siguientes acciones:

1. Desplazamiento al último registro de la base de datos.

2. Posicionamiento en condiciones de poder ser insertado un nuevo registro.

3. Creación de un nuevo registro.

4. Introducción en los campos de los nuevos valores de las variables.

5. Cerrar la actualización.

Para completar o aclarar cualquier cuestión relacionada con la programación de bases de datos en VB, consultar [2], [3] y [4].

3.4.2 Interacción cliente-base de datos

Para simplificar la tarea de desarrollo de la aplicación cliente a partir del servidor se ha considerado la opción de utilizar un objeto *ADO* (ver [2] y [3]). De esta manera resulta enormemente sencillo el seguimiento de la base de datos por parte del cliente, dado que no hay más que enlazar los cuadros de texto con los campos del objeto *ADO* que se quieran mostrar y/o controlar. Mediante un control *Timer*, que no es más que un evento al que asignamos un intervalo de tiempo, transcurrido el cual se desencadena, se consigue actualizar los valores. Esta actualización se consigue con el siguiente código:

```
     Adodc1.Refresh
     Adodc1.Recordset.MoveLast
```

Para el caso estudiado, se fija este valor del intervalo de refresco en 250 milisegundos. Éste será acorde con el tiempo de integración considerado en el experimento base, puesto que debe ser lo suficientemente rápido para detectar posibles cambios en el modelo.

La asignación de nuevos valores de las variables se hace de forma directa, programando el evento que se considere oportuno de los objetos deseados.

Por ejemplo, la sintaxis del código para un deslizador será:

```
Private Sub Slider_MouseUp(Button As Integer, Shift
As Integer, X As Single, Y As Single)

Adodc1.Recordset.Fields(i) = Slider.Value
Adodc1.Recordset.Update
Adodc1.Refresh
Adodc1.Recordset.MoveLast

End Sub
```

Donde:

1. Se asigna el valor del deslizador al campo correspondiente en la tabla de la base de datos.

2. Se aplica la orden de actualización.

3. Se refrescan los valores del objeto para que se encuentren activos.

4. Se desplaza al último registro.

4 CONCLUSIONES

Aprovechando las capacidades de conectividad que ofrece ECOSIMPRO, resulta simple y eficaz utilizar este software como base para trabajar en entornos de programación orientados al desarrollo de aplicaciones SCADA, utilizando como nexo entre el servidor y los distintos clientes una base de datos y las posibilidades de manejo de las mismas que ofrece Visual Basic.

En este trabajo se ha ejemplificado todo el desarrollo mediante dos tanques en serie con recirculación. Un ejemplo como este podría servir de punto de partida para:

a. la creación de una serie de aplicaciones a pequeña escala que representen unidades o procesos simples, los cuales, posteriormente, puedan integrarse de forma sencilla para formar procesos o plantas de mayor complejidad.

b. el desarrollo de aplicaciones web que permitirían tanto un proceso de aprendizaje de los procesos

representados como las el desempeño de las funciones reales de una aplicación SCADA (monitorización, control y supervisión) vía Internet.

Agradecimientos

Deseamos dar las gracias a Pedro Cobas y a Juan Carlos Arroyo de EA International, por su disposición a aclararnos cualquier duda surgida durante la realización de este trabajo y por su inestimable ayuda.

Este trabajo ha sido financiado por el proyecto del Plan Nacional de I+D+I PPQ2001-3643.

Referencias

[1] Bristol, E.H. (1966) "On a new measure of interactions for multivariable process control" *IEEE Transactions on Automatic Control*, volume CA-11, pp. 133-134.

[2] Havolrson, M. (1999) Aprenda Visual Basic Ya 6.0, McGraw-Hill, Madrid.

[3] McManus, J.P. (1999) Bases de datos con Visual Basic 6, Prentice-Hall, Madrid.

[4] Microsoft Press (1998) Microsoft Visual Basic 6.0. Manual del programador, McGraw-Hill, Madrid.

[5] Ogunnaike, B.A., Harmon, R. (1994) Process Dynamics, Modelling and Control, Oxford University Press, New York.

APÉNDICE A CÓDIGO DEL COMPONENTE tanques_en_serie

```
USE CONTROL

COMPONENT tanques_en_serie

PORTS

    IN analog_signal he
    IN analog_signal F2e
    IN analog_signal F1e
    IN analog_signal be
    OUT analog_signal href1
    OUT analog_signal nivel1
    OUT analog_signal href2
    OUT analog_signal nivel2
    OUT analog_signal T1sref
    OUT analog_signal T1sal
    OUT analog_signal T2sref
    OUT analog_signal T2sal


DATA

--Datos de caudales

REAL F1r=0.05   --Caudal de referencia de la
                  corriente entrada al tanque 1 (m3/s)
REAL F2r=0.05   --Caudal de referencia de la
                  corriente entrada al tanque 2 (m3/s)
REAL k1r=0.1    --Constante de proporcionalidad (en
                  e.e.) entre el Fsalida y h en el
                  tanque 1 (m2/s)
REAL k2r=0.1    --Constante de proporcionalidad (en
                  e.e.) entre el Fsalida y h en el
                  tanque 2 (m2/s)
REAL T1r=20     --Temperatura de referencia de F1
                  (ºC)
REAL T2r=60     --Temperatura de referencia de F2
                  (ºC)
REAL T1sr=31.36 --Temperatura de referencia para
                    la salida del tanque 1 (ºC)
REAL T2sr=39.88 --Temperatura de referencia para
                    la salida del tanque 2 (ºC)
REAL br=0.4     --Valores de referencia para la
                  definición del reciclo desde la salida
                  del Tanque 2 al Tanque 1 (m2/s)
REAL gr=0.6


--Datos geométricos de los tanques

REAL At1=0.5    --Área transversal del tanque 1 (m2)
                REAL At2=0.2  --Área transversal
                  del tanque 2 (m2)
REAL hr1=1.361  --Altura del tanque 1 en el estado
                  estacionario (m)
REAL hr2=2.777  --Altura del tanque 2 en el estado
                  estacionario (m)
```

```
DECLS
    REAL F1         --Caudal de la corriente de
                    entrada al tanque 1 (m3/s)
    REAL F2         --Caudal de la corriente de
                    entrada al tanque 2 (m3/s)
    REAL T1         --Temperatura de F1 (ºC)
    REAL T2         --Temperatura de F2 (ºC)
    REAL T1s        --Temperatura de salida del
                    tanque 1 (ºC)
    REAL T2s        --Temperatura de salida del
                    tanque 2 (ºC)
    REAL h1         --Altura de líquido en el
                    tanque 1 (m)
    REAL h2         --Altura de líquido en el
                    tanque 2 (m)
    REAL k1         --Grado de apertura de la
                    válvula de salida del
                    tanque 1
    REAL b          --Valores de definición del
                    reciclo
    REAL g
    REAL z          --Variable auxiliar para el
                    control de la altura en el
                    tanque 1
    REAL y          --Variable auxiliar para el
                    control de la altura en el
                    tanque 2
    REAL x          --Variable auxiliar para el
                    cálculo de T2s
    REAL r          --Variable auxiliar para el
                    cálculo de T1s
    REAL p          --Variable auxiliar para el
                    control de T1s
    REAL m          --Variable auxiliar para el
                    cálculo de T2s
    REAL hr1v       --Perturbación en el nivel
                    de referencia del tanque 1
                    (m)
    REAL hr2v       --Perturbación en el nivel
                    de referencia del tanque 2
                    (m)
    REAL T1sr1      --Perturbación en la
                    temperatura de referencia
                    del tanque 1 (m)
    REAL T2sr1      --Perturbación en la
                    temperatura de referencia
                    del tanque 2 (m)

    INIT

    CONTINUOUS

    h1'=(F1+(b*k2r*sqrt(h2))-(k1*sqrt(h1)))/At1

    h2'=(F2+(k1*sqrt(h1))-(b*k2r*sqrt(h2))-
    (g*k2r*sqrt(h2)))/At2

    x'=((T1s*k1*sqrt(h1))+(T2*F2)-
    (T2s*k2r*sqrt(h2)*b)-
    (g*k2r*sqrt(h2)*T2s))/(At2*hr2)

    T2s=x/h2

    r'=((F1*T1)+(b*k2r*sqrt(h2)*T2s)-
    (k1*sqrt(h1)*T1s))/(At1*hr1)

    T1s=r/h1

    --Control de la altura en el tanque 1

    nivel1.signal=h1
    href1.signal=hr1v
    z=he.signal
    k1=k1r+z

    --Control de la altura en el tanque 2

    href2.signal=hr2v
    nivel2.signal=h2
    y=F2e.signal
    F2=F2r-y

    --Control de la temperatura de salida del tanque 1
    (T1s)

    T1sref.signal=T1sr1
    T1sal.signal=T1s
    p=F1e.signal
    F1=F1r+p

    --Control de la temperatura de salida del tanque 2
    (T2s)

    T2sref.signal=T2sr1
    T2sal.signal=T2s
    m=be.signal
    b=br+m
    g=1-b


END COMPONENT
```