Conexión DDE entre Matlab y Fix Dmacs (software de supervisión, control y adquisición de datos - SCADA) para un controlador difuso

DDE conections between Matlab and Fix Dmacs (supervisory software, control and data acquisition-SCADA) for a difut controlleter

Panayotis S. Tremante M., Mercedes Torres R., José Alonso S.
Departamento de Electrónica y Control. Escuela de Ingeniería Eléctrica. Universidad Central de Venezuela. Apartado Postal 51334. Sabana Grande. Caracas 1050-Venezuela. Tlf: (582) 6053230/3305/3236. Fax: (582) 6053105 e-mail: ptreman@elecrisc.ing.ucv.ve, mtorres@elecrisc.ing.ucv.ve, jalonso@elecrisc.ing.ucv.ve

Resumen

En este trabajo se implementó un Bloque de Control Difuso para el software FIX DMACS de Intellution, utilizado en Supervisión, Control y Adquisición de Datos (SCADA). El Controlador Difuso se diseña con el Fuzzy Logic Toolbox de MATLAB, que permite elaborar y simular el controlador diseñado. La comunicación para el intercambio de información entre ambos paquetes, los cuales se encuentran bajo ambiente Windows, se realiza a través de la conexión DDE. Los pasos en el desarrollo de la conexión DDE son indicados en este trabajo y así, una vez diseñado el Bloque Difuso, puede ser utilizado en aplicaciones industriales.

Palabras Claves: Control Difuso, Control de Procesos Industriales, Automatización, Adquisición de Datos, SCADA.

Abstract

In this work a Block of Fuzzy Control was implemented for Intellution's FIX DMACS, a software system that provides supervisory control and data acquisition (SCADA). The Fuzzy Controller is designed with the Fuzzy Logic Toolbox of MATLAB that allows to elaborate and to simulate the designed controller. The communication for the exchange of information between both software, under Windows environment, are carried out through the connection DDE. The steps in the development of the connection DDE are indicated and, once designed the Diffuse Block, it can be used in industrial applications.

Key words: Fuzzy Control, Industrial Process Control, Automation, Data Acquisition, SCADA.

1 Introducción

El gran auge en la evolución de herramientas computacionales modernas nos permite resolver problemas en el campo de la automatización y control de procesos industriales. En una automatización óptima, se logra la eficiencia y productividad requerida, por lo tanto, es importante la búsqueda de un buen método de control, una adquisición y un manejo de datos de manera eficiente para cada proceso industrial. En este sentido, el uso de paquetes computacionales ya existentes son herramientas para solucionar nuestros problemas.

Un método de control, que en estos últimos tiempo se ha considerado como una alternativa de control en procesos industriales por alcanzar resultados satisfactorios, es el Control Difuso. Algunos procesos industriales se caracterizan por tener un modelo matemático complejo y en muchos casos difícil de obtener, presentando solamente información que está disponible en forma cualitativa, es decir, en términos lingüísticos de funcionamiento. Esta clase de procesos al ser controlados mediante técnicas convencionales de control no logra cumplir de manera efectiva los requerimientos establecidos. El control basado

en lógica difusa ha demostrado ser útil para sistemas altamente complejos que incluyen no-linealidades e incertidumbres en el modelo. El Control Difuso ocupa un lugar importante en los sistemas de control, ya que, se pueden definir reglas de control sin la necesidad de emplear elementos matemáticos complejos, simplemente se definen reglas por medio de palabras de uso cotidiano basado en un conocimiento del sistema. El control difuso elimina los altos contenidos de matemática y física de un proceso, y va directo al nivel en que el sistema trabaja, esto permite aproximarse intuitivamente a la solución de un problema mediante la declaración de reglas. La forma de expresar las reglas de operación mediante palabras permite controlar procesos sencillos con una decena de reglas. Otra ventaja del control difuso, es la fácil modificación que puede llevarse a cabo mediante el cambio de algunas premisas y operaciones, o adicionando reglas, mientras que en un sistema de control convencional, un pequeño cambio requiere la derivación completa de nuevas ecuaciones. (Boverie S., Demaya B., Titli A., 1991), (Wang, Li-Xin. 1994).

Debido a que el Control Difuso se ha considerado como una alternativa de control en el manejo de procesos industriales se han desarrollados herramientas computacionales en el área de la Lógica Difusa, por ejemplo, el Fuzzy Logic Toolbox de MATLAB, es un instrumento eficaz en la concepción y diseño de sistemas inteligentes (Roger Jang J.-S, Ned Guilleym, 1995). También existen otra gran variedad de fabricantes que han desarrollado software y hardware en aplicaciones con Lógica Difusa, fabricantes como Microchip Technology Inc. , Intel, Motorola, National Semiconductor poseen microcontroladores para trabajar con Lógica Difusa, convirtiéndose así en una alternativa más apropiada y económica. (Legg Gary, 1993), (Govind Navin, 1994), (Gullett Ch., Elting D., Kowalczyk R., Fennich M., 1994), (Schreiber R., 1995).

Existen programas utilizados en la automatización de procesos industriales para sistemas SCADA, uno de ellos es el FIX DMACS de Intellution's, un software que proporciona datos en tiempo real y tiene funciones básicas de adquisición y manejo de datos (Intellution1, 1994). Dentro de la adquisición de datos, el FIX DMACS posee la capacidad de extraer los datos de la planta en comunicación directa con los equipos de entrada/salida, la interfase con los equipos se realiza por medio de los drivers de entrada/salida. En el manejo de datos se procesan y manipulan los datos adquiridos; lo que permite, la supervisión del proceso (despliegue gráfico), control, alarmas, reportes y almacenamiento de los datos. Estas características permiten a un operador de planta tener acceso instantáneo a información crítica del proceso y tomar decisiones en línea de forma rápida y efectiva.

Cada día, más y más fabricantes de software descubren lo importante y la ventaja competitiva que es, el intercambio de información entre los diferentes programas de computación. La conexión DDE (Dynamic Data Exchange) es un método de comunicación que usa memoria compartida para intercambiar datos entre aplicaciones en ambiente Windows, (Intellution2, 1994), (Matlab, 1996); programas con la posibilidad de trabajar con esta conexión se convierten en muy versátiles a la hora de necesitar algún componente existente en otro software, como es el caso de nuestro trabajo. Aquí se presenta la conexión entre el FIX y el MATLAB, realizando el diseño del Bloque Difuso en MATLAB, ya que, no se dispone de un Bloque Difuso en FIX.

2 Descripción

El planteamiento del problema surge ante la necesidad de usar un Controlador Difuso que pueda controlar un proceso industrial en tiempo real. El Diagrama de bloque propuesto se muestra en la figura 1

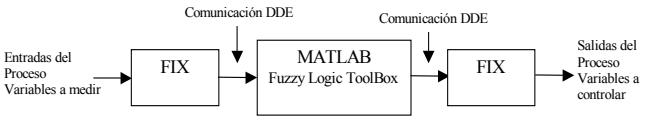

Fig. 1. Diagrama de bloque

Con el software FIX DMACS de Intellution's V6.1 para Windows se adquieren los datos en tiempo real conjuntamente con la instrumentación del proceso, también se pueden obtener datos de equipos remotos o controladores lógicos programables PLC (Programmable Logic Controller) en los cuales se encuentran conectados los sensores o transductores. Dentro de la arquitectura básica del FIX se encuentra la Base de Datos de Proceso (PDB) hecha con Bloques (también llamados *tags*), que son un conjunto de instrucciones destinado a realizar una función de proceso. Una de las funciones es controlar, por lo cual se tiene un bloque PID, pero la falta de un bloque de control con lógica difusa nos condujo a diseñarlo con el Fuzzy Logic Toolbox de MATLAB V5.0, como se observa en la figura 1.

Ambos programas soportan el DDE para obtener datos en tiempo real, por consiguiente, es de gran ayuda utilizar este método de comunicación en la solución del problema.

En la figura 2 se tiene el algoritmo a seguir en la implemetación del controlador difuso, indicando el lugar donde se aplica la comunicación DDE. Se definen las variables de entrada a ser medidas en el proceso y a través, del FIX se adquieren los datos de las variables de interés, con los datos en FIX se pueden mostrar en pantalla (despliegue) o adecuar los datos para enviarlos a MATLAB. Una vez que los datos han sido procesados son enviados a MATLAB vía DDE, los datos recibidos por MATLAB son cargados al controlador difuso, al finalizar la evaluación del controlador se envían de nuevo los datos a FIX con el DDE. Y finalmente los datos recibidos por FIX son manipulados o procesados para llevarlos a las variables de salida del proceso, es decir, a las variables a controlar. Este procedimiento se realiza en forma cíclica y continúa durante todo el proceso de control o al menos que se defina una interrupción deseada en el algoritmo de implementación

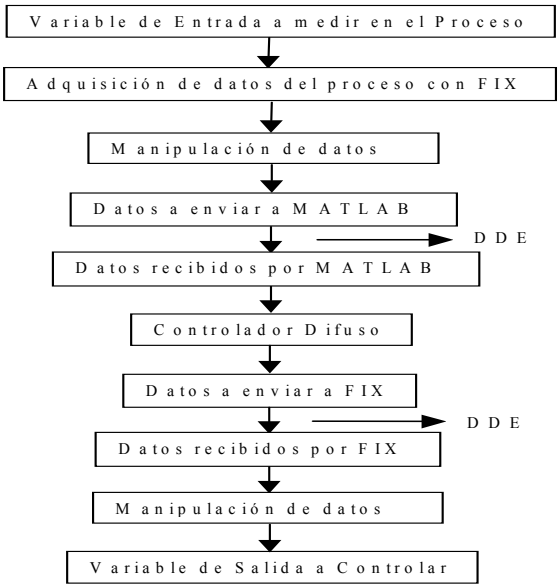

Fig. 2. Algoritmo para la implementación del Bloque Difuso.

3 Controlador difuso

Con el Fuzzy Logic Toolbox de MATLAB se construye el controlador difuso definiendo todos los términos lingüísticos y sus funciones de pertenencia para cada variable lingüística. El Toolbox proporciona dos formas de crear un sistema difuso: por líneas de comandos o por la Interfaz Gráfica del Usuario (GUI – Graphical User Interface). Dentro de cualquiera de las formas escogida para diseñar el controlador, existen cinco fases a realizar que son:

Fusificación de las entradas.
Aplicación del Operador Difuso.
Aplicación del Método de Implicación.
Asociación de las salidas.
Defusificación.

Dentro de los principios básicos de las fases antes mencionadas el Toolbox de Lógica Difusa es muy flexible, permitiendo particularizar el sistema difuso para una aplicación. MATLAB permite sustituir una función predefinidas por una función definida por el usuario, es decir, uno mismo puede realizar su propia función de pertenencia, métodos AND, métodos OR, métodos de implicación, métodos de asociación y métodos de defusificación. Una abierta y fácil modificación del sistema es una de las principales metas del Toolbox.

3.1 Fusificación de las Entradas.

El primer paso es tomar la entrada y determinar su grado de pertenencia en cada uno de los conjuntos difusos correspondientes. La entrada siempre es un valor numérico discreto limitado por el universo de discusión de la variable entrada y la salida es un grado de pertenencia difuso (siempre entre 0 y 1). Así, que la fusificación realmente no es mas que la obtención del valor en una tabla o la evaluación de una función.

3.2 Aplicación del Operador Difuso.

Una vez que las entradas han sido fusificadas, sabemos el grado de cada parte del antecedente que satisface cada regla. Si el antecedente de una regla dada tiene más de una parte, el operador difuso se aplica para obtener un número que represente el resultado del antecedente de esa regla. Este número se aplicará entonces, a la función de salida. Las entradas al operador difuso son dos o más valores de pertenencia de las variables de entrada fusificadas. En los operadores lógicos difusos se pueden realizar las operaciones AND y OR. En el Toolbox de Lógica Difusa, dos métodos para el AND son soportados: el *min*(mínimo) y el *prod*(producto). También para el OR se tiene dos métodos: el *max*(máximo), y el probabilístico OR llamado método *probor*. El método probabilístico OR (también se conoce como la suma algebraica) y se calcula según la ecuación (1):

$$probor(a,b) = a + b - ab \quad (1)$$

3.3 Aplicación del Método de Implicación.

Antes de aplicar el método de implicación, se debe tomar en cuenta el peso de la regla. Cada regla tiene un *peso* (número entre 0 y 1) que se aplica al número dado por el antecedente. Generalmente este peso es 1 y no tiene ningún efecto en el proceso de implicación. Pero de vez en cuando se puede cambiar el peso de la regla a otro valor diferente de 1. El método de implicación define la formación de la

consecuencia (conjunto difuso) basado en el antecedente (un solo número). La entrada del proceso de implicación es un número dado por el antecedente, y la salida es un conjunto difuso. La implicación ocurre para cada regla. Dos métodos son soportados, los mismos del operador AND: el *min*(mínimo) qué trunca la salida del conjunto difuso, y el *prod*(producto) qué escala la salida del conjunto difuso.

3.4 Asociación de las salidas.

La asociación es la unión de las salidas dada por cada regla. Justamente aquí se toman todos los conjuntos difusos que representan la salida de cada regla y se combinan en un solo, en preparación para la defusificación que es el paso final. La asociación sólo ocurre una vez para cada variable de salida. La entrada al proceso de asociación es la lista de las funciones de salida obtenidas en el proceso de implicación de cada regla. El método de asociación es conmutativo, siendo el orden en que se ejecutan las reglas insignificante. Tres métodos son soportados: el *max*(máximo), *probor*(probabilístico), y el *sum* (simplemente la suma de la salida de cada regla).

3.5 Defusificación.

La entrada de la defusificación es un conjunto difuso (la salida de la asociación) y la salida final es un número discreto (crisp), aunque se hayan realizadas tantas fusificaciones como evaluaciones de reglas durante en el proceso intermedio. Así, dado un conjunto difuso que abarque un rango de valores de salida, se necesita obtener al final un número, por eso de un conjunto difuso se llega a una salida discreta. El método de defusificación más popular es el cálculo del centroide que devuelve el centro de área bajo la curva. Hay cinco métodos en el Toolbox: el centroide, bisector, el medio del máximo (el promedio del valor de los máximos del conjunto de salida), el mayor de los máximos, y el menor de los máximos.

4 Implementación

Para la implementación del Bloque Difuso se debe previamente programar la comunicación DDE tanto para FIX como MATLAB, en la sección 4.2 se observa un ejemplo de esta programación. La aplicación se desarrollo en los laboratorios de control distribuido de nuestra Universidad, en procesos de control de temperatura y control de nivel.

4.1 Direccionamiento del DDE.

Los programas que soportan el DDE utilizan la sintaxis ATI que se refiere a: Application Topic Item (ATI). El ATI es un formato estándar de identificación de la información DDE y la descripción de la sintaxis es: **=Application|Topic!Item**. La "Application" es el nombre de la aplicación DDE donde se encuentran los datos, muchas aplicaciones utilizan el nombre del programa. El "Topic" es el nombre del grupo de datos a leer, muy a menudo es el nombre del archivo. El "Item" representa la estructura de los datos a transferir, el nombre del "Item" depende de la aplicación. En el caso del MATLAB el formato ATI es: =MATLAB|ENGINE!Z.

4.2 Ejemplo

El ejemplo siguiente muestra la programación realizada para la comunicación DDE aplicado a un control de temperatura. En la figura 3 se indica la programación en FIX y en la figura 4 la programación en MATLAB.

La programación en el FIX se realizó con el Lenguaje de Comandos el cual es una herramienta para automatizar ciertas operaciones a través, de una serie de instrucciones que solo son compatibles con FIX.

```
DECLARE #RETRIEVE STRING
DECLARE #RESULTADO NUMERIC
&L1
GETVAL =MATLAB|ENGINE!Z #RETRIEVE
STRTONUM #RETRIEVE #RESULTADO
SETVAL FIX:DATO_RECIBIDO_DE_MATLAB.F_CV #RESULTADO
GOTO L1
```

Fig. 3. Programación en FIX

La programación en MATLAB se realizó en un archivo de instrucciones con extensión: **.m** que puede ejecutarse desde MATLAB. Este archivo se encarga de establecer la comunicación DDE con instrucciones que posee MATLAB para trabajar con DDE y a su vez, hace referencia al archivo del controlador difuso previamente diseñado.

```
%Inicialización de Variables
b=0;

%Se carga todas las variables del Controlador Difuso en la matriz (a)
a=readfis('temperatura');

while b~=100
  %Inicializa canal para la conversación con FIX.
   chan=ddeinit('dmdde','data');
   %Solicita el dato a FIX
   data=ddereq(chan,'fix.DATO_ENVIADO_A_MATLAB.f_cv');
   b=data;
   %Retardo para cargar dato en memoria
   pause(.1)
  %Finaliza Conversación
   rc=ddeterm(chan);
  %Evaluación del Controlador Difuso
   Z=evalfis(data,a);
end
```

Fig. 4. Programación en MATLAB

En la figura 3 se observa la declaración de la variable #RETRIEVE como una cadena de caracteres y #RESULTADO como una variable numérica, la variable #RETRIEVE se utiliza debido a que el FIX recibe los datos en forma de cadena de caracteres. La instrucción GETVAL carga en la variable #RETRIEVE el dato enviado por MATLAB llamado Z. En STRTONUM se convierte la cadena de caracteres en un valor numérico. Finalmente con SETVAL se coloca el valor numérico en #RESULTADO en un punto de la base de datos que en el ejemplo esta definido con FIX:DATO_RECIBIDO_DE_MATLAB.F_CV. Con el GOTO se vuelve a ejecutar de nuevo la petición de datos.

En la programación de MATLAB de la figura 4 se observa primero la inicialización de las variables, posteriormente se cargan las variables del controlador difuso diseñado, con la instrucción "readfis" y luego en un ciclo continuo se realiza la petición de los datos a FIX. El formato ATI para FIX es: La "Application" es 'dmdde', el "Topic" es 'data y el "Item" es el nombre en la base datos de FIX.

5 Conclusiones

Los resultados obtenidos con esta aplicación fueron satisfactorios, ya que permitieron relacionar en tiempo real las potencialidades de cada software, e implementar un Bloque Difuso. En este caso, el FIX utiliza las herramientas matemáticas en el desarrollo de un controlador difuso dado por Fuzzy Logic Toolbox, mientras, que MATLAB aprovecha la capacidad en la adquisición y manejo de datos que ofrece un programa en sistemas SCADA.

El aspecto más importante es mostrar las virtudes y metodología en la implementación de un bloque difuso usando el intercambio de información DDE para la conexión de paquetes computacionales existentes, en la solución inmediata de un problema de control industrial.

Los aportes introducidos por las herramientas computacionales en la realización de controladores difuso para ser aplicados a diferentes procesos simples, permite verificar las estrategias de control y evaluar el desempeño de la solución. Al establecer el control de un proceso con un computador personal y la instrumentación existente en campo conduce a una gran versatilidad y confiabilidad a la hora de implementar un controlador difuso.

6 Referencias

Boverie S., Demaya B., Titli A., 1991. "Fuzzy Logic Control compare with other Automatic Control approaches", 30th IEEE Conference on Decision and Control, Brighton-England, , Pag. 1212-1216.

Govind Navin, 1994. "Fuzzy Logic Control with the Intel 8XC196 Embedded Microcontroller", Application Note, Intel Corporation, Pag. 1-11.

Gullett Ch., Elting D., Kowalczyk R., Fennich M., 1994. "Intel Fuzzy Logic Tool Simplifies ABS Design", Application Note AP-700, Intel Corporation, Pag. 1-6.

Intellution1, 1994. "Fix Dmacs Student Guide", Intellution Inc.

Intellution2, 1994. "Manual Advanced Tools", Intellution In

Legg Gary, 1993. "Microcontrollers embrace fuzzy logic", EDN, Pag. 100-109.

Matlab, 1996. "Application Program Interface Guide", The Math Works Inc.

Roger Jang J.-S, Ned Guilleym, 1995. "Fuzzy Logic Toolbox For Use with MATLAB", The Math Works Inc.

Schreiber R., 1995. "Air Flow Control Using Fuzzy Logic", Application Note AN600, Microchip Technology Inc., Pag. 1-24.

Wang, Li-Xin. 1994. "Adaptive Fuzzy Systems & Control: Design & Stability Analysis". Prentice Hall. 240 P. ISBN 0-13-099631-9.