ENSEÑANZA Y APLICACIÓN DE CAN (CONTROLLER AREA NETWORK)

Lenin Lemus[1], Juan Miguel García[1], José Luis Bayo[1]

[1]*E.U.I (Escuela Universitaria de Informática de la Universidad Politécnica de Valencia.)*
e-mail: lemus@disca.upv.es, juagarg2@eui.upv.es, jobamon1@eui.upv.es

Resumen: Hoy en día, la mayor parte de las aplicaciones que se implementan utilizan redes de ordenadores. En el caso particular de los entornos industriales una de las redes más utilizadas es CAN (Controller Area Network). Esta es la razón por la cual estamos convencidos que a los alumnos de ingenierías técnicas y superiores les será de gran provecho conocer CAN para poder desarrollar aplicaciones basadas en este protocolo.

En esta ponencia se presenta un simulador que permite comprender el funcionamiento de CAN, a la vez que permite simular de forma gráfica el desarrollo de aplicaciones industriales basadas en sensores y actuadores inteligentes. El simulador está implementado en Java, debido a que es un lenguaje portable y que además permite generar aplicaciones que pueden ser ejecutadas de forma local o bien de forma remota mediante el uso de un navegador de Internet. El objetivo primario que se persigue con este simulador es la enseñanza de CAN.

La organización de la ponencia es la siguiente: En la primera sección se presenta la introducción, en la segunda sección se presenta de forma breve las principales características de CAN. En la tercera sección se presenta el simulador implementado. Finalmente, en la cuarta y quinta sección se presentan las conclusiones y la bibliografía respectivamente.

1.- INTRODUCCIÓN.

Una de las principales características de una aplicación industrial a tomar en cuenta en su desarrollo es el hecho de que se trata de una aplicación distribuida, entendiendo por aplicación distribuida aquella en la que los datos y los métodos que actúan sobre ellos se encuentran dispersos en un área geográfica determinada. En la mayoría de los casos de aplicaciones industriales distribuidas, la información es generada por sensores, las acciones necesarias son ejecutadas por actuadores y el procesamiento de la información es llevado a cabo por algoritmos ejecutados mediante micro-controladores o por ordenadores personales.

De cualquier forma, el trasiego de la información se lleva a cabo, conectando mediante redes de área local. Dada la naturaleza de la aplicación puede ser necesario la utilización de mecanismos que detecten anomalías, funcionamientos erróneos y eviten al máximo situaciones que puedan resultar catastróficas e inseguras para los seres humanos, e incluso para los mismos elementos de la aplicación, por lo que resulta necesario que las aplicaciones cuenten con mecanismos de tolerancia a fallos. En resumen se puede decir que cuando se desarrolla una aplicación industrial el tipo de red que se utilice debería cumplir con las siguientes características básicas:

i. Proporcionar una alta fiabilidad.
ii. Tener un coste bajo tanto de instalación como de mantenimiento.
iii. La instalación y mantenimiento deben ser fáciles de llevar a cabo.

Desde la década de los años sesenta, tanto empresas como universidades y organismos de estandarización se han dedicado a desarrollar e implementar redes especializadas para uso industrial, de estos esfuerzos nacieron los denominados buses de campo. Sin embargo, para el caso concreto de la industria automotiva, las empresas automotivas desarrollaron a principios de los años ochenta protocolos propios, entre los cuales destacan: CAN, VAN, ABUS, CCD, J1850, etc..

CAN [1] fue desarrollado por R. Bosch GmbH. En 1985 Bosch e Intel colaboraron para diseñar los circuitos integrados que implementan el protocolo CAN. La primera empresa en utilizar CAN fue Mercedes Benz en 1992, en los automóviles de tipo "Mercedes class S", poco tiempo después BMW, Porsche y Jaguar entre otros empezaron a utilizarlo en sus vehículos. A pesar de que CAN fue desarrollado para aplicaciones automotivas, sus características básicas han sido explotadas para desarrollar aplicaciones industriales [2], convirtiéndose en una de las redes más utilizadas en el entorno industrial.

En esta ponencia se presenta un simulador que permite al alumno comprender como funciona CAN, a la vez que permite el desarrollo de aplicaciones industriales basadas en sensores y actuadores.

2.-CARACTERÍSTICAS BÁSICAS DE CAN.

La descripción detallada del protocolo "Controller Area Network" ha sido normalizado por la "International Standard Organization" ISO, las normas establecidas son: ISO-11898 [3] e ISO-11519 [4]. A continuación se enumeran las principales características del protocolo CAN.

i. Es un protocolo orientado a mensajes, utiliza bits dominantes y recesivos, en el caso de CAN el '0' es el bit dominante y el '1' es el bit recesivo. Cada uno de los mensajes CAN debe tener asociado un Identificador (ID).
ii. La contienda para ganar el acceso al medio de comunicación se realiza escribiendo en el bus el ID del mensaje que se desea transmitir.
iii. Define dos formatos de trama denominados CAN2.0A y CAN 2.0B, la diferencia básica es el número de bits que se utilizan para el identificador.
iv. Implementa los niveles Físico y de Enlace de datos del modelo "International Standard Organization/Open Systems Interconnections" (ISO/OSI).
v. Efectúa la detección de errores a nivel de y a nivel de bit.
vi. Puede transmitir datos con tasas de transmisión que van de 128 Kbps hasta 1Mbps.
vii. Cuando se realiza la comunicación, CAN utiliza la codificación NZR (Non Return to Zero) con bit de stuffing.
viii. La conexión física al medio de comunicación se realiza mediante transceptores, esto permite aislar el protocolo CAN del tipo de medio de transmisión utilizado, como medio físico para transmisión de mensajes, CAN acepta cualquier medio en el que se pueda implementar el uso de bits recesivos, los más comunes son: par trenzado, cable coaxial, fibra óptica.

3.- SIMULADOR DE APLICACIONES CAN

Para facilitar la enseñanza del funcionamiento de CAN decidimos desarrollar un programa con las siguientes características:

i. El programa debe permitir visualizar como funciona CAN
ii. Debe poder ser ejecutado tanto en plataformas UNIX tanto como en plataformas Windows 9X.
iii. Debe poder ser ejecutado de forma remota.
iv. Debe poder ser distribuido en forma de librería para que puedan ser utilizados de forma fácil los componentes desarrollados.

En base a estas características decidimos utilizar el lenguaje de programación Java [5], debido principalmente a:

i. La existencia de compiladores y entornos de desarrollo gratuitos
ii. Facilita el desarrollo de aplicaciones que pueden ser ejecutadas en diferentes plataformas de forma remota, mediante los applets.
iii. Para el caso de la reutilización java tiene definidos los "Java-Beans" que nos permiten encapsular y distribuir componentes que pueden ser reutilizados.

Funcionamiento Básico:

El usuario debe ejecutar desde un navegador (Netscape, Internet Explorer) o desde el appletviewer, el applet de la simulación de dos CPU's conectadas a través de CAN. Al ejecutar el applet aparecerá la aplicación mostrada en la figura 1.

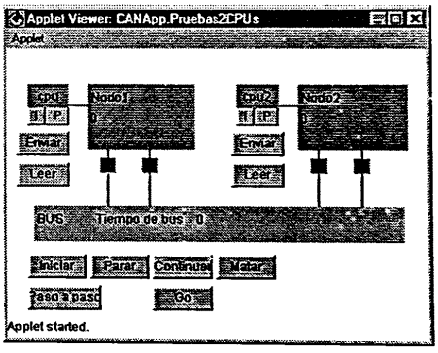

Figura 1. Red CAN constituida por dos nodos CAN.

El primer paso es configurar las máscaras que permiten a los nodos filtrar las tramas que deben procesar. Para ello basta colocarse sobre el Nodo1 o el Nodo 2 y oprimir el botón "Info". La siguiente figura 2 muestra la ventana que aparece. En la tabla Mascaras, se editan los ID que debe aceptar este nodo. El cuadro de diálogo también muestra el estado actual de las colas de E/S.

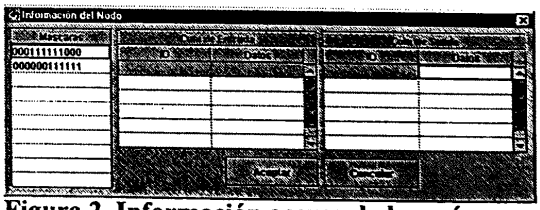

Figura 2. Información acerca de las máscaras del Nodo1 y el contenido de las colas de transmisión y recepción.

Una vez que han sido programadas las máscaras, se cierra la ventana de la pantalla "Información del nodo", y en el applet se oprime el botón de "Iniciar" que pone el estado lógico del bus a nivel alto.

Con esto el bus empezará a mostrar los tiempos de bit que están transcurriendo, como lo muestra la figura 3. A partir de este momento los nodos competirán para enviar sus mensajes por el bus.

Para transmitir un mensaje, basta con oprimir el botón "Enviar" de la CPU, que aparece en el Applet, con esta acción automáticamente se encola en la cola de transmisión del nodo el mensaje y cuando el bus está libre, el nodo entra en contienda por obtener el control del bus, si lo gana, entonces transmite el mensaje, la transmisión de un mensaje se observa visualmente de la siguiente forma: El color que indica el tiempo de bit en el Nodo transmisor se escribe con color rojo, mientras que el tiempo de bit de los receptores se resalta con un fondo gris. En el bus se muestra con un fondo rojo. Ver figura 4.

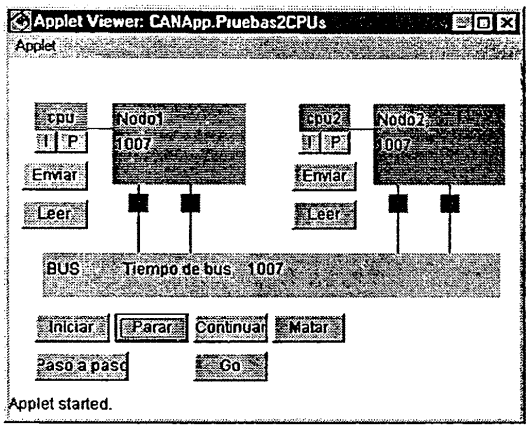

Figura 3. Simulación de la aplicación, tanto el bus como los nodos muestran el tiempo de bit actual.

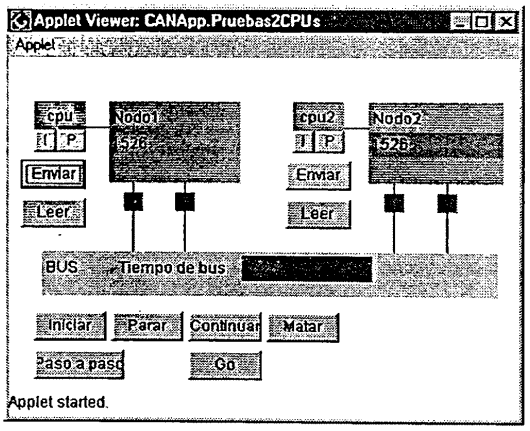

Figura 4. Transmisión de un mensaje.

Con el botón "I", la CPU, genera de forma continua mensajes. Con el botón "P" detenemos la generación de mensajes. Si queremos desencolar mensajes recibidos de alguna CPU se debe presionar el botón "leer" asociado con a ella. Para tener un mayor control de la simulación se ha incluido una ejecución "Paso a paso", que detiene la simulación cada vez que un mensaje es enviado a los nodos destino. Con "Go", la simulación continua hasta el siguiente envío de mensaje.

Para el control de bus está el botón de "iniciar" (ya comentado anteriormente) para su puesta en marcha o reiniciación; "parar" detiene la simulación; "continuar" reanuda la simulación en el estado que quedó en la parada y "matar" detiene la simulación de bus.

Para ayudar al usuario, cada vez que se efectúa un evento importante (emisión, recepción, etc.) se muestra en la consola java el tipo de evento acaecido, el elemento donde se generó y el tiempo de bus del instante del suceso.

4.- CONCLUSIONES:

Se han desarrollado 4 JavaBeans, correspondientes al BUS, los transceptores, el Nodo-CAN y una CPU. Estos Componentes permiten crear aplicaciones Java o Applets, que simulan redes de área local industriales.

En nuestro caso permiten al usuario ver como opera el BUS CAN. Al ser componentes del tipo "Java Bean" su distribución se facilita, mediante el uso de los ficheros de tipo JAR.

Los Applets se pueden ubicar en servidores WEB y ejecutarlos de forma remota mediante un navegador de internet.

La distribución en formato jar permite a cualquier programador hacer uso de los beans para desarrollar sus propias aplicaciones.

Para obtenerlos enviar un correo electrónico a esta dirección: lemus@disca.upv.es

5.- BIBLIOGRAFÍA

[1] Philips Semiconductors. "CAN Specification Version 2.0". Philips Semiconductors Unternehmensberei der Philips GmbH Burchardstrasse 19,D-20095 Hamburg Germany.

[2] Wolfhard Lawrenz. "CAN System Engineering. From Theory to Practical Applications".Springer Verlang 1997.

[3] International Standard ISO-11898. ISO Reference Number ISO 11898:1993(E). First Edition 1993/11/15.

[4] International Standard ISO-11519-2. ISO Reference Number ISO ISO-11519-2:1994(E). First Edition 1994/06/15.

[5] James Gosling, Bill Joy and Guy Steele. "The Java Language Specification". Addison-Wesley 1996.