



ACCESO REMOTO A DATOS EN PLATAFORMA ARM VÍA ETHERNET

(ARM remote data access through Ethernet)

Hernández, Fidel; Celorio, Carlos

Universidad de Pinar del Río, Cuba, CP 20100.

Hernández, Fidel E.

Investigador, Universidad de Pinar del Río, calle Martí final, Pinar del Río, Cuba.
Teléfono: 5348779362. Correo electrónico: fidel@tele.upr.edu.cu.

Celorio, Carlos A.

Investigador, Universidad de Pinar del Río, calle Martí final, Pinar del Río, Cuba.
Teléfono: 5348779362.

RESUMEN

El Grupo de Investigación (GIDAM), de la Universidad de Pinar del Río, trabajó en un proyecto orientado al diagnóstico de sistemas electromecánicos (aerogeneradores), para lo cual se dispuso de un sistema embebido de tecnología ARM, que cuenta con datos de mediciones almacenados en una memoria local USB, pero sin garantizarse el acceso remoto a los datos. Este acceso es importante para monitorear el estado de los componentes electromecánicos desde una computadora situada centralmente en el parque eólico. Como objetivo de la investigación, se pretendió implementar el acceso remoto a los datos almacenados en la memoria USB vía Ethernet, utilizando el servidor web Boa (en el ARM). En particular, se publicó una página web dinámica para el monitoreo de los datos almacenados. Las mediciones realizadas fueron almacenadas en ficheros XML y se hizo uso de la biblioteca LIBXML2, en combinación con el lenguaje C, para extraer los datos de cada variable, tratarlos, y mostrar el resultado en la página web. Se utilizó el programa Digi ESP, a través del cual se logró hacer la compilación y simulación del código escrito en C. Al finalizar, se obtuvo el monitoreo y acceso remoto a los datos almacenados en la memoria USB.

Palabras claves: Sistema embebido, Servidor web Boa, acceso remoto a datos.

ABSTRACT

The Research Group for Advanced Machine Diagnosis, at the University of Pinar del Río, Cuba, works on a project approached to machine monitoring and diagnosis, in particular, wind turbines. To do so, a platform based on ARM embedded technology was used and data from sensors was stored into an USB memory; no remote data access was provided. It is important to provide remote data access because data processing must be performed by a computer placed at a central area in the wind farm. The goal of this work is to implement remote data access, via Ethernet, in an embedded system based on ARM using the web server Boa. Specifically, a dynamic web page was created and sensor measurements are stored in XML files and library



LIBXML2 and software C are used for reading, processing and publishing data. Program Digi ESP was used for compiling and simulating the program written in C. Then, the monitoring and remote access of data stored in memory USB is attained.

Keywords: Embedded system, web server Boa, remote data access.

1. Introducci n

A partir de la creaci n del Grupo de Investigaci n para el Diagn stico Avanzado de Maquinarias (GIDAM), de la Universidad de Pinar del R o, desde enero de 2000, y la materializaci n de un conjunto de resultados investigativos, se comenz  a efectuar el diagn stico por vibraciones en diferentes empresas del territorio (por ejemplo, en la Empresa Provincial de Piezas de Repuesto, Empresa de Agromec nica y Empresa Provincial C rnica), aplicando tecnolog a de bajo costo y fundamentos avanzados de an lisis.

Esto se produjo gracias al desarrollo de dos sistemas de medici n de vibraciones (Diagnos V1 y Diagnos V2), los cuales permit an, el primero de ellos, captar, almacenar y procesar se ales de vibraci n siguiendo un enfoque offline; y el segundo realizaba estas tareas de manera online, posibilitando la vigilancia continua y remota de vibraciones, corrientes y tensiones el ctricas, temperatura, entre otros. A estos resultados se le adiciona un paquete en desarrollo de herramientas cl sicas y avanzadas de an lisis de vibraciones para el diagn stico.

GIDAM ha venido trabajando en varios proyectos para el desarrollo de tecnolog a de diagn stico industrial. Uno de ellos se relaciona con su participaci n en la red iberoamericana CYTED-SEDIPRE, que tiene como objetivo obtener un sistema embebido orientado al mantenimiento predictivo, en particular, al diagn stico de sistemas electromec nicos, para lo cual se puso a disposici n de sus miembros, un sistema embebido basado en tecnolog a ARM.

Hasta el momento se ha establecido el sistema de modo tal que se digitaliza un grupo apreciable de se ales de sensores que almacenan los datos en una memoria local USB. Esto no ayuda en cuanto a la idea de obtener dichos datos de manera remota, es decir, desde un lugar distante.

Problema de la investigaci n: para el estado actual de la plataforma de trabajo ARM, no se tiene acceso remoto a los datos almacenados en una memoria USB conectada a dicha plataforma. Este acceso puede ser importante, ya que una de las aplicaciones espec ficas del sistema de diagn stico propuesto involucra a los aerogeneradores, para lo cual es necesario recibir informaci n  til del estado de los componentes electromec nicos, en una computadora situada centralmente en el parque e lico, sin subir hasta las turbinas.

Objetivos de la investigaci n: realizar el estudio y poner en pr ctica un servicio http en la plataforma ARM, v a Ethernet y utilizando el servidor web (Boa), que viene

instalado con el Sistema Operativo Embebido del ARM; así como la implementación de una página web dinámica en la plataforma ARM, mediante la cual sea posible obtener los datos de manera remota y el monitoreo de la condición electromecánica de los aerogeneradores.

2. Elementos necesarios para establecer un servicio web

2.1. Uso de CGI para el trabajo con los datos en el lado del servidor.

En la aplicación a desarrollar es necesario mostrar de forma dinámica en una página web los datos actuales de las variables a medir en los aerogeneradores. Estos datos deben estar almacenados en ficheros xml (a caracterizar más adelante), de los cuales se toma el último valor de algunas variables, así como el valor rms de algunas señales.

Estas tareas no las puede realizar el servidor web, debido a que esa no es su función, sino solamente el trabajo con la web y los recursos relacionados con la misma. Dada esta necesidad, se debe usar un programa CGI (Common Gateway Interface: interfaz entre aplicaciones externas y servicios de información) (Introducción..., 2009).

En "la red" existen aplicaciones CGI desarrolladas en C, C++, Fortran, Perl, Tcl, Visual Basic, AppleScript y cualquier shell de UNIX. En la figura 1 se muestra un gráfico de la función del CGI en combinación con el servidor web para el servicio al cliente.

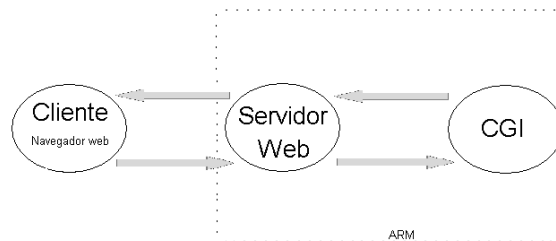


Figura 1. Esquema en bloques de la comunicación a nivel de programas entre Cliente y Servidor, interviniendo el CGI.

Si se tienen en cuenta las opciones de la tarjeta ARM en cuanto a lenguajes de programación, se obtiene que sólo se puede usar C o C++. En el siguiente apartado se hace una breve caracterización del lenguaje C.

2.2. Lenguaje C como herramienta para programar el CGI.

Los elementos de un programa en C son los siguientes:

Comentarios	Variables locales
Inclusión de archivos	Flujo de sentencias
Main()	}
{	Definición de funciones creadas por el programador usadas en main()



Cada elemento se define como sigue (Lenguaje..., 2009):

- Comentarios: se identifican porque van entre diagonales y asterisco.
- Inclusión de archivos: consiste en llamar a la o las bibliotecas donde se encuentran definidas las funciones de C (instrucciones) que se están utilizando en el programa.
 - main(): en C, todo está constituido a base de funciones. El programa principal no es la excepción, main() indica el comienzo de la función principal del programa, la cual se delimita con llaves.
 - Variables locales: antes de realizar alguna operación en el programa, se deben declarar las variables a utilizar en el programa.
 - Flujo de sentencias: es la declaración de todas las instrucciones que conforman el programa.
 - Definición de funciones creadas por el programador utilizadas en main(): finalmente, se procede a definir el contenido de las funciones utilizadas dentro de main(). Éstas contienen los mismos elementos que la función principal.

Debido a que, básicamente, la gran diferencia entre C y C++ es la orientación a objetos de este último, más la poca demanda de la investigación sobre cualquiera de los dos lenguajes, se ha escogido el lenguaje C para realizar esta aplicación de CGI.

Así como C, es usado para el programa que va a funcionar como CGI, se usan también los ficheros del tipo XML para el almacenamiento de los datos, lo cual ofrece ventajas incomparables.

2.3. Ficheros con extensión XML para almacenar los datos.

XML (eXtensible Markup Language) fue creado al amparo del World Wide Web Consortium (W3C), organismo que vela por el desarrollo de WWW partiendo de las amplias especificaciones de SGML; este lenguaje consiste en un sistema para definir, validar y compartir formatos de documentos en la web (Melián, 2009).

Ventajas del lenguaje XML:

- Sintaxis simple.
- Legible para las personas.
- Muy parecido al HTML.

En el interior de los ficheros XML se presentan los datos de manera estructurada, o sea, en forma de árbol. El trabajo estructurado es extremadamente potente, ya que mediante un parser (analizador) de XML es muy fácil la labor para cualquier aplicación.

El lenguaje XML debe ser interpretado por algún programa, de modo tal que los datos que se encuentran en la memoria USB sean tratados hasta mostrarlos en una

página web por el servidor. Libxml2 es justo la herramienta que se necesita para el trabajo con los ficheros XML.

2.4. Caracterización de la biblioteca LIBXML2.

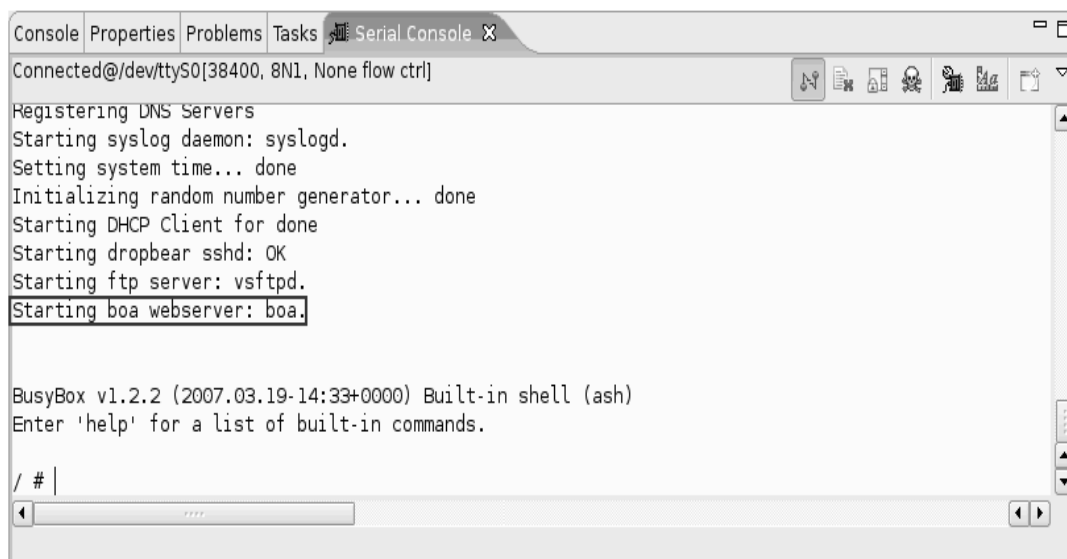
LIBXML2 es una biblioteca que implementa un analizador sintáctico XML y permite hacer uso de XML en aplicaciones. Libxml2 ofrece una importante lista de posibilidades, es decir, con LIBXML2 se puede:

- Analizar un documento.
- Extraer texto de un elemento específico.
- Añadir un atributo, así como el contenido de éste.
- Extraer el valor de un atributo.
- Crear nuevos documentos, así como introducirle nodos atributos y valores.

Mezclados el lenguaje C y las funciones de LIBXML, se obtienen los datos de los ficheros XML; ahora, es necesario que estos lleguen hasta el cliente (mediante una página web), y para esto se debe utilizar el servidor web Boa (que está instalado en el kit), el cual toma el código HTML y lo envía al cliente (Programación..., 2009).

2.5. Boa, el servidor web.

Boa es el servidor web que ofrece el kit ARM, el cual viene por defecto instalado en la tarjeta. Como se ve en la figura 2, es inicializado por el sistema cuando levanta. Lo observado en la figura 2 es la consola serie del programa Digi ESP, el cual ha sido proporcionado por el fabricante para el trabajo con el módulo ARM.



```
Console Properties Problems Tasks Serial Console X
Connected@/dev/ttyS0[38400, 8N1, None flow ctrl]
Registering DNS Servers
Starting syslog daemon: syslogd.
Setting system time... done
Initializing random number generator... done
Starting DHCP Client for done
Starting dropbear sshd: OK
Starting ftp server: vsftpd.
Starting boa webserver: boa.

BusyBox v1.2.2 (2007.03.19-14:33+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

/ # |
```

Figura 2. Consola serie de Digi ESP, mostrando cómo el Servidor web Boa es inicializado por el sistema operativo cuando arranca.



3. CGI en el ARM

Despu s de haber analizado la importancia de los CGI para los servidores web, cabe destacar el objetivo de su uso en el ARM: tratar los datos de las variables almacenadas en la memoria USB y devolver resultados al servidor web.

El c digo HTML de la p gina web es el resultado del CGI, el argumento de cada printf usado en el c digo en C, se trata de puro lenguaje HTML. El servidor Boa toma el c digo HTML y lo env a al explorador, que se encarga de interpretar el c digo HTML y mostrar la p gina. Cada segundo una funci n implementada en JavaScript hace al servidor la petici n de ejecuci n del CGI, de esta forma se repite la ejecuci n del CGI descrita anteriormente.

Dentro de la funci n de JavaScript implementada se destaca la l nea que expresa lo din mico de la p gina web propuesta:

```
setTimeout("location.href='http://10.2.132.134/cgi-bin/test'", 1000);
```

 donde test es el ejecutable que contiene el compilado del CGI en C.

Para que el servidor, a solicitud de la petici n, ordene la ejecuci n del CGI, se debe especificar la ruta de acceso para los CGI; mediante la directiva del servidor, ScriptAlias, se le especifica dicha ruta de acceso, como se muestra a continuaci n:

```
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
```

La p gina que muestra el resultado de esta aplicaci n ocupa un espacio de 32 kiB, el cual no resulta comparable con lo que ocupan las p ginas m s peque as encontradas normalmente (m s de 300 kiB), casi sin im genes y s lo texto), lo que significa que las transferencias de datos por la red no son considerables.

3.1 Programaci n del CGI.

Para la programaci n del programa al lado de all  en el servidor, se utiliz  el esquema representado en la figura 3. En la funci n principal (main()) se env a, primeramente, (usando el comando printf) parte del c digo HTML; la cabecera de la p gina y la declaraci n de la tabla.

Luego, se comienza con el trabajo sobre los documentos XML a trav s de la funci n parseDoc() con la cual se realiza la validaci n de errores, es decir, se verifica la existencia del documento a analizar, y dentro de  ste, los nodos principales.

Despu s de haber quedado hecha la validaci n se comienza a recorrer el interior del archivo, a trav s de la funci n parsevalores(), con la que se recorre cada uno de los nodos hijos (valor y fecha), depositando su contenido en las variables correspondientes.

Cada vez que se recorre un documento extrayendo los valores de cada variable, se imprime mediante printf el código HTML correspondiente a una fila de la tabla; lo correspondiente a una variable.

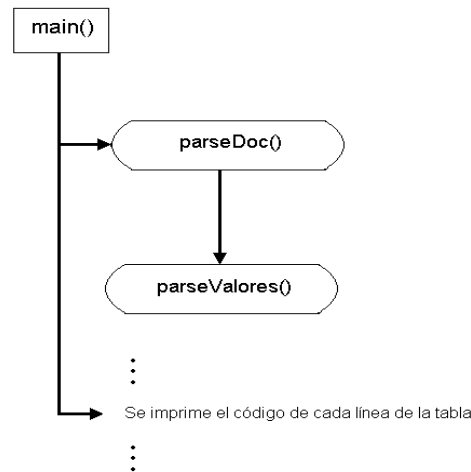


Figura 3. Esquema en bloques del código del CGI.

Luego de haber realizado el código y haberlo compilado en Digi ESP, el fichero binario pasa a la tarjeta hacia el destino seleccionado, /usr/lib/cgi-bin. Para comprobar todo en el navegador web de una PC conectada a un hub de Ethernet, al cual también está conectado el kit de ARM, se colocó `http://10.2.132.134/cgi-bin/test`, obteniéndose los resultados esperados y mostrados en la figura 4.

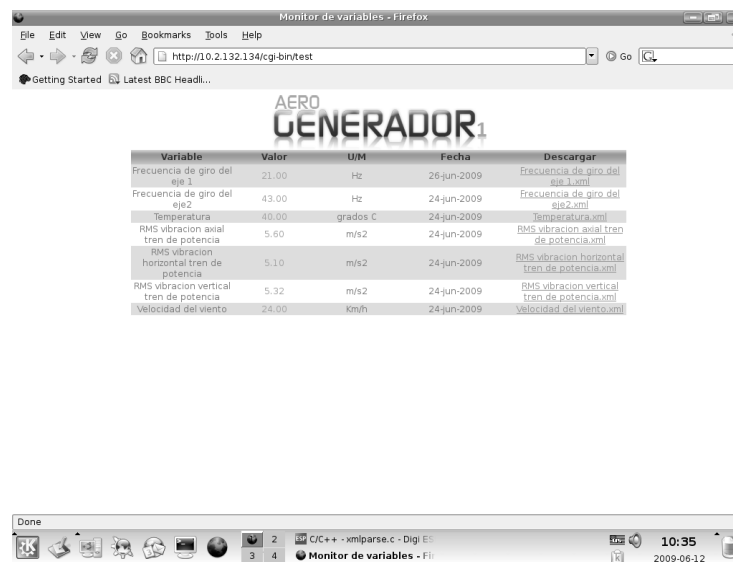


Figura 4. Web del Aerogenerador 1, como resultado de la implementación del CGI y el servidor.



Conclusiones

Se puede concluir que se logró implementar el servicio HTTP en la plataforma ARM, contribuyendo a la obtención de un sistema de monitoreo real de la condición electromecánica de los aerogeneradores, lo cual proporcionó el acceso remoto a datos almacenados en la memoria USB, conectada al ARM.

Referencias bibliográficas

Introducción a la Interfaz CGI. Documento en línea. Disponible en:

http://www.ulpgc.es/otros/tutoriales/New_cgi/new_cgi1.htm. Consultado el 28 de septiembre de 2009.

Lenguaje C. Documento en línea. Disponible en:

<http://orbita.starmedia.com/~prog201eq1/AGO-20-1.HTM>. Consultado el 28 de septiembre de 2009.

Melián M. XML el nuevo lenguaje universal. Documento en línea. Disponible en:

<http://www.bibliociencias.cu/qsdl/collect/eventos/index/assoc/HASH0104/f016d031.dir/doc.pdf>. Consultado el 28 de septiembre de 2009.

Programación en el entorno GENOME. Documento en línea. Disponible en:

<http://www.calcifer.org/documentos/librognome/xml.html>. Consultado el 28 de septiembre de 2009.