Revista Iberoamericana de Automática e Informática Industrial
CEA IFAC comité español de automática
RIAI
ISSN: 1697-7912. Vol. 2, Núm. 2, Abril 2005, pp. 30-35
http://riai.isa.upv.es

LABORATORIO DE AUTOMÁTICA VÍA INTERNET (LAVI)

Carlos A. Zuluaga Toro, Carlos G. Sánchez Toro y Elkin A. Rodríguez Ortiz

caz@logos.upb.edu.co, cgsanchez@upb.edu.co, elkinrodriguezo@yahoo.com
Facultad de Ingeniería Electrónica, Universidad Pontificia Bolivariana
Medellín - Colombia

Resumen: Se presenta el diseño y desarrollo de una plataforma de software que permite realizar pruebas remotamente sobre una planta de control lineal. La plataforma se formula a partir de la configuración de un sistema cliente/servidor, en el cual Apache administra las páginas web. El proceso de petición y respuesta de usuario es ayudado por programas escritos en Perl, los cuales realizan la comunicación con la interfaz existente en Matlab, para que ésta se encargue del control de la planta del laboratorio de Automática. Adicionalmente, un sistema administrador de bases de datos, MySQL, dirige el control de acceso de los estudiantes e investigadores.

Palabras claves: Laboratorio de enseñanza, TCP/IP, Sistemas de administración de bases de datos, Sistemas de control lineal, Lenguajes de programación, Control de acceso, Control remoto

1. INTRODUCCIÓN

El avance de la tecnología de la informática y las telecomunicaciones ha impulsado el desarrollo de sistemas para la transmisión de instrucciones de control desde una ubicación remota. Estos sistemas promueven grandes ventajas que han sido aprovechadas en laboratorios de distintas universidades del mundo, lo que los convierte en una realidad apremiante para la educación (S. H. Chen, *et al*., 1999; Dormido, 2000; Ríos, *et al*., 2003).

El laboratorio de Automática de la Universidad Pontificia Bolivariana ha cumplido con el desarrollo de proyectos encaminados a la adaptación de las herramientas que utiliza, y con los avances tecnológicos y de innovación que aparecen en el mundo de la ingeniería. Es por eso que, en la nueva etapa de modernización y adaptación del laboratorio que se lleva a cabo en la actualidad y, como una primera aproximación, se crea el presente trabajo que tiene como fin aprovechar las posibilidades que ofrece el control remoto, empleando Internet, en la educación. Esto es, brindar nuevos espacios curriculares que optimicen el uso de los equipos y la disponibilidad horaria del laboratorio, así como la colaboración con otras instituciones que no posean estos recursos.

La plataforma de software creada en el curso de este proyecto permite, a estudiantes o investigadores con conexión a Internet, hacer uso de la planta lineal con la que cuenta el laboratorio de Automática (llamada Movilab) a través de una de sus interfaces de control creada en Matlab (interfaz conocida como Realmatlab), para el desarrollo de las pruebas de

control que empleen referencia por Software. En el Laboratorio de Automática Vía Internet (LAVI) es posible, por medio de una interfaz gráfica de usuario (GUI), modificar a criterio de los estudiantes, los parámetros del control, para finalmente, luego de realizada la acción de control obtener los resultados de la prueba para el posterior análisis.

2. ESTRUCTURA DEL LABORATORIO REMOTO

El laboratorio remoto tiene una estructura de hardware y software que ya existía, y otra estructura de software, que constituye la plataforma de LAVI.

2.1. Laboratorio físico

Se maneja una planta básica: un motor de CD, con unos módulos que permiten crear sistemas multivaribles tanto continuos como discretos y probar los manejando dicha planta. Estos módulos, conocidos como Movilab, adicionaron una mejor comprensión de los conceptos, introduciendo una explicación gráfica del recorrido de las señales dentro de un sistema de control en tiempo real

Esta planta trabaja con una plataforma gráfica, creada por Osorio (2001), que permite el diseño personal de sistemas de control usando la sintaxis de Matlab, lenguaje altamente empleado en la investigación y la ingeniería. Es a partir de esta herramienta creada en Matlab y actualmente conocida como "Realmatlab" que se ha desarrollado este proyecto.

El "Realmatlab" funciona actualmente en la versión 5.3 de Matlab y cuenta con la ayuda de un "control ActiveX (Control Activex Multimedia de Entrada/Salida de Puertos) para la transmisión y recepción de la información en tiempo real hacia la planta MoviLAB. Está estructurado a partir de una serie de funciones que cumplen tareas específicas como las de construcción de la interfaz gráfica, puesta en funcionamiento del control ActiveX, manejo de rutinas para los diferentes parámetros de control, puesta en funcionamiento de la planta y almacenamiento del resultado obtenido" (Osorio, 2001).

2.2. Estructura de LAVI

LAVI funciona con una configuración de red que sigue el modelo cliente/servidor utilizando tecnología Internet. "Este modelo está basado en la idea del servicio, en la que el proceso cliente envía un mensaje solicitando un determinado servicio a un

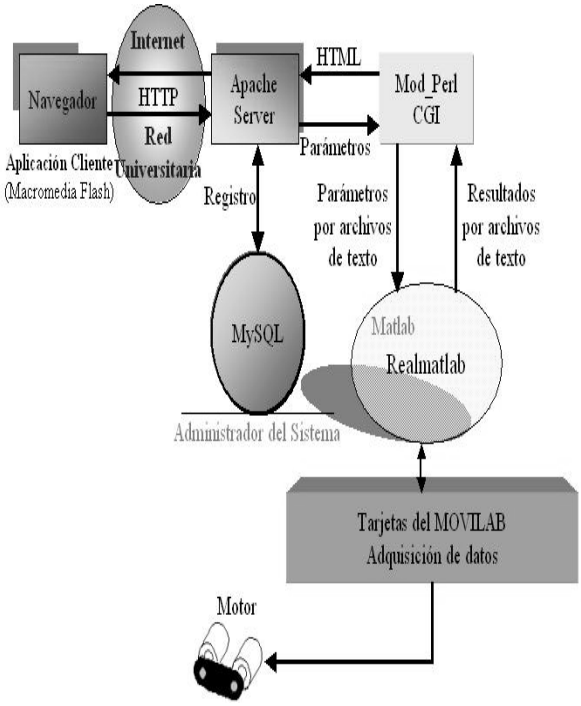

Figura 1. Esquema del Laboratorio de Automática Vía Internet, LAVI.

servidor, y éste envía uno o varios mensajes con la respuesta o servicio" (Laurie y Laurie, 1999). El esquema detallado de la comunicación puede verse en la figura 1.

LAVI se basa en la configuración de un sistema servidor, *Apache* (Versión 2.045), el cual administra las páginas *web* que el usuario remoto utiliza en sus pruebas de laboratorio y, a su vez, interpreta las peticiones del usuario y retorna las respuestas que la planta le entrega. El proceso de petición y respuesta es ayudado por *scripts CGI* (*Common Gateway Interface* o Interfaz Común de Pasarela) escritos en Perl, los cuales realizan la comunicación con la interfaz ya existente en Matlab (Realmatlab) y la manipulan. Por otro lado la plataforma cuenta con un sistema administrador de bases de datos, MySQL, el cual permite dirigir el proceso de registro y control de acceso de los usuarios que quieren entrar al sistema remotamente.

Así, el usuario podrá acceder a la página del laboratorio ingresando la dirección adecuada, pero debe previamente cumplir con los requisitos de registro y control de acceso, administrados por una base de datos creada para el correcto funcionamiento del laboratorio remoto. Luego de superar el control de acceso, podrá visualizar la interfaz gráfica de control creada en Macromedia Flash MX, que le permitirá ingresar los parámetros de la práctica y

enviarlos al servidor, activando el programa que inicializa la planta, realiza el control y finalmente devuelve los resultados y gráficas de la prueba.

3. SERVIDOR WEB

Se basa en el Protocolo HTTP (Protocolo de Transferencia de Hiper Texto), el cual es un sencillo protocolo cliente-servidor que articula los intercambios de información entre los clientes y los servidores Web. Está soportado sobre los servicios de conexión TCP/IP (Protocolo de Control de Transmisión/Protocolo Internet). Un proceso servidor escucha en un puerto de comunicaciones TCP (por defecto el 80) y espera las solicitudes de conexión de los clientes web. Una vez que se establece la conexión, el protocolo TCP se encarga de mantener la comunicación y garantizar un intercambio de datos libre de errores.

El Servidor Web Apache es un servidor robusto, de código libre y modular, que facilita los servicios con los protocolos TCP/IP. La versión 2.0.45 contiene modificaciones en el diseño para un trabajo óptimo con Windows (LAVI trabaja con Windows 98), ya que Apache fue diseñado inicialmente para trabajar en sistemas Unix y es en éstos donde muestra un mejor desempeño (Kabir, 2002).

4. INTEGRACIÓN PERL/APACHE

La meta principal del proyecto de integración Apache/Perl, en los criterios de Bekman y Maceachem (2003), era traer todo el poder del lenguaje de programación de Perl dentro del servidor Apache. Esto se transformó en el desarrollo de mod_perl. Mod_perl es un paquete de software que incrusta un intérprete Perl en el servidor Apache, dando al desarrollador acceso a la API (Interfaz de Programación de Aplicaciones) de Apache desde Perl.

El ambiente de mod_perl es bastante diferente del ambiente CGI. Debido a que, bajo mod_perl, el intérprete Perl es parte del proceso del servidor y no hay necesidad de que éste lo comience o detenga para cada solicitud. Y, ya que los *scripts* mod_perl son realmente módulos de Perl que extienden la funcionalidad del servidor, no hay necesidad de volver a compilar el código Perl en cada solicitud. Esto hace de la ejecución del código de Perl sumamente rápida dentro del ambiente de mod_perl.

LAVI utiliza las ventajas de mod_perl 2.0 en el control de acceso y autorización de usuarios (aprovechando algunos de los muchos módulos que se han escrito especialmente para el ambiente Apache/mod_perl.); en el uso de variables globales (para que Matlab actúe como un proceso servidor); y en el aumento de velocidad de ejecución de los *scripts* CGI tradicionales.

5. COMUNICACIÓN MATLAB-PERL

El manejo remoto de la planta lineal del laboratorio, es posible gracias al desempeño de un objeto COM (*Component Object Model* o Modelo de Componentes de Objeto) creado para permitir la utilización del programa Matlab desde un archivo ejecutado en Perl. Luego de la apertura de Matlab es posible introducir cualquier tipo de comando y llamar las funciones que activan el programa Realmatlab en el computador del laboratorio. Todo ello desde la ventana del navegador de un estudiante ubicado en cualquier parte del mundo.

Para que las aplicaciones como Matlab permitan a otro software acceder a sus capacidades, deben exponerle sus servicios internos; en otras palabras, deben ser programables. Proporcionar esta programabilidad es la meta de Automatización (originalmente conocido como Automatización de OLE (*Object Linking and Embedding*), o OLE Automation). Con esta tecnología, Matlab y el *script* de Perl pueden funcionar como componentes COM, aquél actuando como servidor, y éste como cliente (Chappell, 1996).

LAVI utiliza la herramienta Automatización de ActiveX (familia de tecnologías orientadas a objeto que tienen una raíz común, COM) para la comunicación entre el lenguaje Perl y el software que controla la planta lineal, *Realmatlab*. Así se logra que los parámetros de la práctica, elegidos por el usuario remoto, sean procesados por Matlab y luego los resultados puedan ser presentados por el navegador.

5.1. La API de Matlab

Matlab, como lo establece The Mathworks (2002), proporciona una API para interactuar con datos y programas externos al ambiente de Matlab. Las funciones soportadas por la API incluyen, entre otras, la de establecer relaciones cliente/servidor entre Matlab y otros programas de software (Aplicaciones Cliente/servidor). Dentro de esta función se encuentra el sorporte ActiveX, y más específicamente,

automatización ActiveX (ActiveX Automation). Para usar Matlab como un servidor de automatización, se invoca con el nombre del objeto ActiveX de Matlab, es decir, "Matlab.Application".

La interfaz ActiveX de automatización para Matlab soporta varios métodos, entre los cuales está el método 'Execute', el cual toma una cadena de un comando como un argumento, que puede ser cualquier instrucción que normalmente se teclearía en la ventana de comandos (command window); el resultado contiene cualquier salida que se habría mostrado en la ventana de comandos como resultado de ejecutar la cadena, incluyendo los errores. Y cualquier figura generada, se despliega en la pantalla como si el comando fuera ejecutado directamente desde el computador local.

5.2. Integración de Matlab con Win32::OLE

El módulo Win32::OLE permite a un programa Perl actuar como un controlador ActiveX. Para crear un objeto servidor se necesita crear un objeto Perl para representar el servidor. Así aunque el programa es un controlador o cliente ActiveX, contendrá objetos que representan servidores ActiveX.

LAVI crea un objeto COM mediante el modulo de Perl Win32::OLE, que permite la apertura del programa Matlab durante todo el tiempo que se encuentre en funcionamiento el servidor Apache por medio de una variable global, dejándolo listo para ser utilizado en cada prueba. Su código es el siguiente:

```
# Se carga el módulo OLE de Perl
  use win32::OLE;
# se define $Mlab como una variable global
  use vars qw($Mlab);
# se crea el objeto COM en la variable $Mlab
  $Mlab=Win32::OLE->new('matlab.Application');
```

Otro *script* de Perl se encarga de capturar los parámetros ingresados en la GUI, enviarlos a los archivos ejecutables de Matlab e inicializar la planta del laboratorio (*$Mlab->Execute("lavi.m");*), que realiza la acción de control y almacena el resultado en el servidor.

6. CONTROL DE ACCESO Y AUTORIZACIÓN DE USUARIOS

En todo servidor web en el cual se quiera restringir el acceso a algunos recursos, es necesario realizar un control de acceso de usuario. Por ello, cuando un estudiante quiere acceder a la interfaz que le permite realizar la práctica, primero se ve obligado a introducir su nombre de usuario y contraseña; si son correctos, se procede a verificar si el usuario está autorizado para realizar la práctica en esa fecha y a esa hora. Si no lo está, se le informa su fecha real de la práctica, y en caso contrario, se le concede el permiso al estudiante para acceder al recurso solicitado, a través de una sesión segura de *cookies*. Dicha sesión cuenta con varias medidas de seguridad para impedir que alguien no autorizado pueda acceder a la práctica.

6.1. Control de acceso con MySQL.

En LAVI se emplea el sistema administrador de bases de datos (DBMS, Database Management System) MySQL para el control de acceso. Apache actúa como un usuario de MySQL (llamado 'httpd') y por medio de él se verifican las tablas de la base de datos de LAVI. Apache se conecta al servidor de la base de datos por medio de unos módulos de Perl y realiza sus consultas; MySQL consulta sus bases de datos y retorna las solicitudes al servidor Apache.

6.2. Comunicación con la base de datos.

Debido a la popularidad de Perl, fue sólo cuestión de tiempo antes de que las personas empezaran a usar Perl para interactuar con bases de datos. DataBase Interface (DBI, o interfaz de base de datos) proporcionó a los programadores un conjunto estándar de funciones y variables para acceder cualquier base de datos. La interfaz habla con un driver específico para la base de datos (Figura 2). Esta parte de la arquitectura se llama el database driver (DBD, o driver de la base de datos). El driver es el último vínculo hacia el verdadero servidor de la base de datos. Existen módulos DBD's de Perl para Oracle, SyBase, MySQL, y otras más (Kabir, 2002).

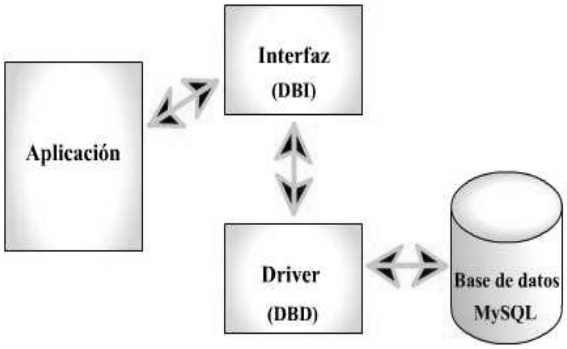

Figura 2. Arquitectura de una aplicación de Perl interactuando con MySQL.

6.3. Sesiones seguras a través de cookies.

El módulo Apache::AuthTicket proporciona el control de acceso basado en *tickets* o boletos. Para su funcionamiento fue necesario realizar unos pequeños cambios para personalizar la aplicación para LAVI, tales como la forma HTML para el ingreso, la autorización de acuerdo al horario de la práctica, una bandera que indica si un usuario está o estuvo realizando la práctica y otros detalles pequeños.

El módulo certifica contra la base de datos la primera vez que el usuario se conecta. Después de una satisfactoria validación de la identidad del usuario y su fecha de práctica, el usuario es marcado con un *ticket* para el uso de accesos subsiguientes. Este *ticket,* que no es más que un *cookie* HTTP, consiste en el tiempo, el nombre de usuario, el tiempo de expiración del *cookie*, una versión confidencial (secret version), y una firma criptográfica. Hasta que expira, el ticket puede ser usado para realizar cualquier control con cualquiera de las referencias, cuantas veces se quiera. Además, este sistema es bastante seguro porque la dirección IP del usuario está incorporada en la firma criptográfica. Si el ticket fuera interceptado, entonces un atacante tendría que robar la dirección IP del usuario para poder usar el ticket.

7. APLICACIÓN CLIENTE

La aplicación en la parte del cliente, navegador del estudiante o investigador, consiste en una película interactiva desarrollada con MACROMEDIA FLASH MX, que toma las ventajas que ofrece esta herramienta en animación, control y manejo de eventos y comunicación externa con otras aplicaciones por medio de su lenguaje *ActionScript.* La película envía las variables que el usuario estableció para la práctica a un *script* de Perl. Se responsabiliza de hacer las verificaciones de las variables, y para ello envía mensajes de error a través de una función de JavaScript; de ejecutar una animación mientras se realiza el control; de controlar los tiempos en la práctica; y de cargar las imágenes del resultado del control pertinente.

8. REALIZACIÓN DEL CONTROL

8.1. Registro.

Cuando un usuario ingresa por primera vez al

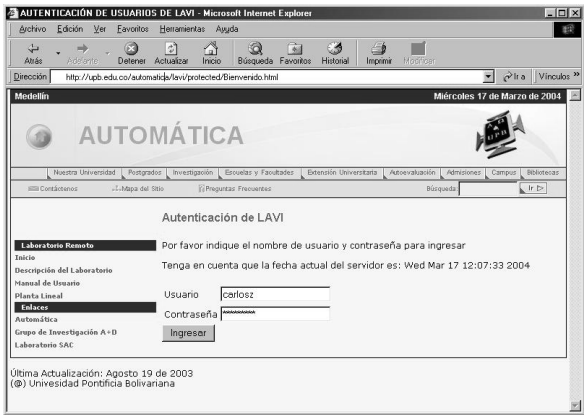

Figura 3. Página de registro de usuario.

controlador de la planta lineal, aparecerá una ventana de registro (Figura 3).

En ella se encuentra la fecha y hora actual del servidor. Se debe ingresar el nombre de usuario y contraseña para el control de acceso. Luego, aparecerá la ventana gráfica que controla la planta lineal (Figura 4).

8.2. Determinar control.

Se debe ingresar uno a uno los valores de los parámetros y las constantes de control en las casillas gráficas (Control P, PI o PID; sus respectivas constantes, período de muestreo, tiempo a controlar, señal de referencia: senoidal, paso, tren de pulsos, diente de sierra o pulso). Luego se envían al servidor, activando el programa que inicializa la planta, realiza el control y finalmente devuelve los resultados de la prueba.

Figura 4. Ventana gráfica para realizar las pruebas remotas.

Figura 5. Ventana gráfica luego de finalizada la prueba.

8.3. Resultado del control.

Al culminar la acción de control aparecerá una imagen con la primera gráfica del resultado de la prueba (Figura 5) y se despliega un menú con las opciones de visualizar la gráfica ampliada y descargar el resultado obtenido en un archivo de Matlab para su posterior análisis.

9. CONCLUSIONES

LAVI permite el desarrollo de las prácticas llevadas a cabo por los estudiantes de automática, en el control de la planta lineal del laboratorio, usando referencia por software, ya que responde correctamente en el transporte de la información y la puesta en funcionamiento del sistema, proporcionando una interfaz segura que puede ser adaptada en caso de que sea necesario, a diferentes aplicaciones académicas similares.

Este proyecto brinda nuevos espacios curriculares con herramientas aplicadas a Internet que dinamizan el ambiente académico y posibilitan la realización de prácticas más rápidamente, con experimentación continua desde cualquier lugar, para estudiantes e investigadores en sitios remotos donde los recursos pueden ser escasos.

En el desarrollo de actividades de investigación del ámbito académico, los programas o software de libre acceso juegan un papel clave, ya que permiten adelantar esta clase de trabajos disminuyendo costos y proporcionando un respaldo de toda una comunidad que está detrás de esta tecnología. Así pues, LAVI aprovechó dichas ventajas con la utilización de aplicaciones como Apache Web Server, MySQL, Perl, Mod_Perl y módulos como Apache::AuthTicket, Win32::OLE, DBI y DBD::MySQL. Esto, sin desmeritar la importancia de programas claves como Matlab y Macromedia Flash.

RECONOCIMIENTOS

Los autores agradecen al grupo de investigación en Automática y Diseño A+D de la Universidad Pontificia Bolivariana de Medellín, por acoger y promover esta propuesta.

REFERENCIAS

Bekman, Stas y Maceachem, Doug. (2003). *Mod_Perl 2.0 Developer's Guide*. 91p. (http://perl.apache.org).

Chappell, David. (1996). *Understanding ActiveX and OLE*. Microsoft Press. p.1-105, 205-235

Dormido, S., Sánchez, J. y Morilla F., 2000 Laboratorios virtuales y remotos para la práctica a distancia de la Automática, sesión plenaria. En: *Actas de las XXI Jornadas de Automática*, Sevilla 18-20 de septiembre de 2000.

Kabir, Mohammed J. (2002). *Apache Server 2 Bible*. New York: Ed. Hungry Minds.

Laurie, Peter y Laurie, Ben. (1999). *Apache: The Definitive Guide*. New York.: O'Reilly & Associates.

Osorio C., Marisol. (2001). Ambiente Interactivo Para Trabajo Experimental en Tiempo Real Usando Matlab. *Tesis de especialización en Automática. Universidad Pontificia Bolivariana.*

Ríos B., Addison y Puente A., Edgar. Diseño e Implementación de un Laboratorio de Control a Distancia. 2da. En: *Conferencia Iberoamericana en Sistemas, Cibernética e Informática CISCI 2003* Agosto 2003, Florida EE.UU. (www.iiisci.org/cisci2004/Pastconference/Program2003.asp).

S. H. Chen, R. Chen, V. Ramakrishnan, S. Y. Hu, Y. Zhuang, C. C. Ko and B. M. Chen. Development of remote laboratory experimentation through Internet. En: *Proceedings of the 1999 IEEE Hong Kong Symposium on Robotics and Control, Hong Kong*, pp. 756-760, July 1999 (http://vlab.ee.nus.edu.sg/publication.html).

The Mathworks (2002). *Matlab, The Language of Technical Computing. External Interface manual, version 6.*