Revista Iberoamericana de Automática e Informática industrial

ISSN: 1697-7912. Vol. 8, Núm. 1, Enero 2011, pp. 89-99

www. revista-riai.org

CtrWeb: Una herramienta de programación para telecontrol de sistemas físicos educativos

Jesús Salido * **Antonio Lillo** ** **Óscar Déniz** * **Mª Gloria Bueno** *

* *Depto. Ing. Eléctrica, Electrónica, Automática y Comunicaciones, Universidad de Castilla-La Mancha (UCLM), Pº de la Universidad 4, 13071, Ciudad Real, España (e-mail: Jesus.Salido, Oscar.Deniz, Gloria.Bueno{@uclm.es})*

** *Escuela Superior de Informática (UCLM) (e-mail: alillocrr@gmail.com)*

Resumen: En este trabajo se presenta una arquitectura software distribuida que proporciona una API *(Application Programming Interface)* de código abierto basada en *Java RMI (Remote Method Invocation)* para la programación, con fines educativos, del control secuencial de prototipos físicos remotos. Los prototipos físicos se construyen mediante el sistema de prototipado de la firma fischertechnik^{TM}. La programación y acceso al prototipo físico se realiza de forma compartida por distintos desarrolladores empleando una estrategia de turnos de tiempo gestionada desde *Moodle*, un LMS *(Learning Management System)* abierto de libre distribución.

Palabras Clave: telecontrol, laboratorio remoto, educación a distancia, automatización, control secuencial.

1. INTRODUCCIÓN

En la última década se ha producido una gran penetración de las tecnologías de la información y comunicaciones (TIC) en el ámbito educativo de los estudios de Ingeniería. Entre las aportaciones más atractivas destacan las posibilidades que se abren para el aprendizaje a distancia (Aktan *et al.*, 1996). Este aprendizaje a distancia puede cubrir no sólo los conceptos teóricos sino también los sujetos a estudio práctico. En la bibliografía especializada se pueden consultar numerosos trabajos realizados sobre el empleo de las TIC en el ámbito de la enseñanza práctica de Ingeniería realizada tradicionalmente en un laboratorio. En este sentido se han publicado revisiones que constituyen estudios esclarecedores sobre el tema (Dormido, 2004; Ma and Nickerson, 2006). Como fruto de dichos trabajos se ha llegado a establecer cuatro grandes categorías de contextos de trabajo:

- *Laboratorios tradicionales*. Son los típicos laboratorios donde el alumno trabaja directamente *in situ* con dispositivos físicos reales.
- *Laboratorios remotos*. En éstos no existe acceso físico directo a los dispositivos del laboratorio sino a través de una interfaz remota de usuario conectada mediante una red de comunicaciones (local o de área extensa).
- *Laboratorios virtuales*. Las experiencias se realizan sobre modelos matemáticos que sustituyen al sistema físico real en estudio. En este caso dicho sistema se sustituye por un programa de computador encargado de proporcionar el realismo requerido en los experimentos.
- *Laboratorios híbridos*. Engloban todas las variantes posibles obtenidas al combinar las alternativas anteriores.

A pesar de todos los trabajos realizados no se ha llegado a un consenso sobre qué metodología produce mejores resultados, existiendo para cada una de ellas tanto defensores como detractores. Como fruto de los estudios y trabajos desarrollados se han identificado una serie de puntos fuertes y debilidades (Dormido, 2004; Aliane, 2008) que conviene tener presente al decidir aplicar las TIC en la enseñanza práctica a distancia de Ingeniería. Dichas fortalezas y debilidades se resumen a continuación:

PUNTOS FUERTES

- *Disponibilidad temporal, espacial y de recursos*. El material es accesible por los estudiantes durante más tiempo, en muchas más localizaciones y con menores restricciones en cuanto a su uso. En muchas ocasiones incluso se plantea ampliar el acceso a estudiantes de otros cursos y titulaciones para que los recursos sean empleados por más usuarios.
- *Mejora del rendimiento*. La amortización del equipamiento tiene un plazo mucho menor debido al uso más intensivo de los recursos. Además al aumentar las posibilidades para compartir los equipos, éstos pueden emplearse por un número mayor de estudiantes.
- *Posible evaluación en línea del proceso de aprendizaje*. El estudiante puede obtener una realimentación más inmediata sobre el cumplimiento de los objetivos del programa formativo.

PUNTOS DÉBILES

- *Aumento del coste de diseño e implantación de laboratorios*. A los costes tradicionales del equipamiento de un laboratorio deben sumarse tanto los asociados al software como al hardware de proceso y comunicaciones.
- *Necesidades de seguridad y fiabilidad*. Es preciso proporcionar nuevos mecanismos que garanticen la seguridad y fiabilidad del acceso a los recursos compartidos.
- *Limitaciones metodológicas*. La naturaleza del proceso formativo a distancia puede hacer inviable la interacción física real con el objeto del estudio y por tanto la realización de ciertas experiencias.

Publicado electrónicamente: 04/01/2011
DOI:10.4995/RIAI.2011.01.11

- *Nuevas necesidades de formación*. Tanto los instructores como los estudiantes deben adquirir formación para el empleo correcto y eficiente de las nuevas tecnologías.

En este artículo no se plantea un análisis exhaustivo de todos los trabajos relacionados con los laboratorios virtuales y remotos, muchos de los cuales aparecen recogidos en excelentes esfuerzos recopilatorios (Candelas and Sánchez, 2005; Grupo de Educación en Automática, CEA-IFAC, 2009). Sin embargo, el estudio bibliográfico realizado por los autores permite extraer una serie de conclusiones respecto a los trabajos analizados y su relación con *CtrWeb*:

- El ámbito de aplicación de los trabajos sobre laboratorios virtuales y remotos se centra principalmente en los sistemas de *control continuo* (Dormido *et al.*, 2000; Casini *et al.*, 2003; Domínguez *et al.*, 2005; Gillet *et al.*, 2005; Valera *et al.*, 2005; Zuluaga *et al.*, 2005; Aliane *et al.*, 2007; Dormido *et al.*, 2008; Macías and Méndez, 2008; Costa *et al.*, 2010; Guzmán *et al.*, 2010) y son pocos los que se ocupan de sistemas con *control secuencial*. Entre estos últimos se encuentran trabajos en el ámbito académico como webPLC (Rodríguez and Neira, 2001), (Gasa *et al.*, 2005), Web-LABAI (Cruz de la *et al.*, 2010), el Laboratorio de Automatización Virtual (GENIA - Univ. Oviedo, 2010) y herramientas comerciales entre las que se destaca Automation Studio™ de Famic Technologies Inc.
- Las arquitecturas empleadas son de tipo cliente-servidor en las que el controlador se implementa del lado del servidor. El cliente es habitualmente una aplicación web en la que se puede simular el comportamiento del sistema en estudio variando los parámetros del controlador y supervisar la ejecución remota de dicho control sobre la planta real. Este tipo de arquitectura es compartido tanto por los sistemas de control continuo como los de control secuencial analizados.
- Los entornos que ofrecen acceso a una planta remota presentan realimentación visual del estado de la planta mediante cámaras de vídeo que en su mayoría integran acceso IP.
- La programación del servidor correspondiente a los distintos laboratorios virtuales y remotos suele realizarse partiendo de entornos comerciales como Matlab/Simulik y NI LabVIEW, aunque en algunos casos también se recurre a lenguajes de propósito general como C/C++ (Aliane *et al.*, 2007; Costa *et al.*, 2010). Para la aplicación cliente la preferencia es el empleo de Java por su facilidad de integración con los navegadores web y su carácter multiplataforma.
- Los laboratorios desarrollados ofrecen utilidades para gestionar el acceso de múltiples usuarios, siendo muy destacable el desarrollo de sistemas como *eJournal* integrados en *eMersion* (Gillet *et al.*, 2005) para la compartición de información y colaboración entre usuarios. Sin embargo son escasos los esfuerzos de integración de soluciones dentro de los *sistemas de gestión de aprendizaje* (LMS) de las instituciones en los que se implantan (García and Rallo, 2005; Sancristobal *et al.*, 2008).

En este artículo se presenta una herramienta de código abierto que permite la programación remota completa del *controlador secuencial* de un sistema físico empleado en prácticas docentes de laboratorio para materias relacionadas con la Ingeniería de Sistemas y Automática. *CtrWeb* está concebida inicialmente como una herramienta para la programación de *sistemas secuenciales* que no ha sido probada para el control de *sistemas continuos* debido a las limitaciones del hardware del controlador FT empleado.

La herramienta proporciona una *interfaz de programación de aplicaciones* (API) en Java y su integración con el *sistema de gestión de aprendizaje* (LMS) *Moodle* (Moodle Org, 2009) empleado en la institución académica donde los autores desarrollan su actividad docente. El empleo de Java aporta ventajas notables como la portabilidad a distintas plataformas, la facilidad de integración en aplicaciones web y la utilización de RMI como mecanismo transparente de comunicación entre la aplicación cliente y el servidor. Respecto a otros trabajos, *CtrWeb* presenta una diferencia sustancial ya que su arquitectura es distribuida y el controlador se implementa en el lado del cliente. La comunicación con el servidor se establece cuando se invoca a las funciones de comunicación con los dispositivos de E/S de la planta. Además todas las tecnologías empleadas tanto en el lado del cliente como del servidor son abiertas.

El planteamiento de *CtrWeb* ha tenido en cuenta algunas de las principales limitaciones de los laboratorios remotos puestas de manifiesto en estudios realizados al respecto (Dormido, 2004; Aliane, 2008). En esta dirección se han previsto mecanismos para la gestión de *timeouts* y en general al tratamiento de situaciones de emergencia o errores de programación que provoquen la no gobernabilidad del sistema.

Este artículo se organiza del modo que se indica a continuación. En primer lugar (sec. 2) se presenta la motivación del trabajo situándolo en su contexto y ámbito de aplicación. En la sección 3 se describe el alcance y el conjunto de objetivos de *CtrWeb*. Las siguientes secciones (secs. 4, 5 y 6) profundizan en los detalles técnicos de implementación de *CtrWeb*: componentes, tecnologías empleadas y arquitectura software. La exposición se completa con una sección de discusión y conclusiones del trabajo presentado (sec. 7) en la que se describe los resultados de la experiencia de los usuarios. Dicha sección se concluye comentando los puntos fuertes y las limitaciones observadas en el trabajo, así como propuestas para superar éstas últimas.

2. MOTIVACIÓN DE LA HERRAMIENTA CTRWEB

El contexto donde se ha desarrollado este trabajo es la Escuela Superior de Informática (ESI) de la Universidad de Castilla-La Mancha (UCLM) en la que desde el curso 1998/99 se viene impartiendo asignaturas ligadas a la Ingeniería de Sistemas y Automática (ISA). En la Tabla 1 se proporciona información sobre dichas asignaturas tanto relativa al marco académico (titulación, curso y carácter) como a la carga docente que suponen (nº de horas de teoría y práctica).

Tabla1. Asignaturas de ISA en la ESI-UCLM

Asignatura	Titulación	Nº hrs. (T/P)
Automatización Industrial	4º ISI, (Obl.)	60/30
Cibernética Aplicada	3º ITIS, 3º ITIG, (Opt.)	45/15

ISI=Ing. Superior en Informática

ITIS= Ing. Téc. en Informática de Sistemas

ITIG= Ing. Téc. en Informática de Gestión

Desde su inicio las asignaturas mencionadas se plantearon con un énfasis especial en sus aspectos prácticos. Este enfoque se fue acentuando con el paso de los años al comprobar la motivación de los estudiantes con el trabajo realizado en los laboratorios. En las prácticas de laboratorio se emplean maquetas (ver Fig.1) construidas con componentes comercializados

por fischertechnik^{TM} (fischertechnik, 2010) que facilitan la implicación del alumno durante las sesiones de laboratorio. En el caso de la asignatura *Automatización Industrial* las prácticas se centran en los *sistemas secuenciales* o *dirigidos por eventos* (p.ej. célula de estampación, robot de soldadura, control de acceso a parking, cruce controlado por semáforos, etc.). En el caso de *Cibernética Aplicada* el sistema empleado es un robot educativo controlado mediante *paradigma basado en comportamientos* (Salido, 2009).

(a) Máquina estampadora (b) Robot móvil educativo

Figura 1. Ejemplo de prototipos físicos empleados

Los prototipos construidos mediante componentes fischertechnik (FT) presentan los elementos característicos de todo sistema electromecánico automatizado:

- Sensores que proporcionan información de realimentación sobre el estado del sistema (interruptores mecánicos, interruptores reed, fototransistores, fotorresistencias, termistores, potenciómetros, etc.).
- Actuadores responsables de los cambios de comportamiento del sistema (motores CC, electroválvulas, electroimanes, avisadores acústicos y luminosos, etc.).
- Controlador electrónico programable que «decide» sobre las acciones instantáneas más convenientes. Además el controlador FT (*Intelligent Interface* ref. 30402, o *ROBO Interface* ref. 93293) incorpora la etapa de potencia necesaria para la conexión directa de los actuadores del sistema sin necesidad de electrónica adicional.
- Elementos mecánicos tanto de soporte como de transmisión de potencia (ejes, bridas, engranajes, poleas, cadenas, etc.)

El control de los prototipos físicos FT se puede realizar mediante dos modos de trabajo: *on-line* y *autónomo*. En el *modo on-line* (ver Fig. 2) el *programa principal de control* se ejecuta en un computador convencional (PC) conectado mediante conexión serie al controlador FT. En éste último se ejecuta un programa de servicio *(FT server)* que forma parte de su *firmware*. En este modo de funcionamiento el programa principal de control realizado por el usuario se puede implementar en cualquier lenguaje de programación de alto nivel y propósito general (Java, C/C++, C#, Python, etc.) que respete el protocolo de comunicación serie impuesto por el controlador FT (RS-232, USB o *wireless*, dependiendo del modelo concreto de controlador). En este sentido uno de los trabajos más completos para la programación *on-line* de los controladores FT es el realizado por U. Müller (Müller, 2008). Con base en dicho trabajo, se elaboró material (Salido, 2006) para las prácticas de laboratorio correspondientes a las asignaturas mencionadas anteriormente, donde se proporciona información general para la programación *on-line* de un controlador FT (*Intelligent Interface*) e incluso una biblioteca de funciones en lenguaje C para la programación de dicho controlador.

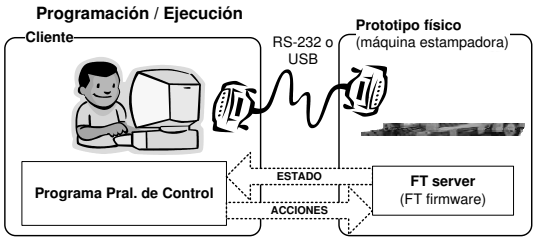

Figura 2. Esquema de trabajo en *modo on-line*

El *modo on-line* es apropiado para el control de prototipos físicos estáticos, entendiendo como tales aquellos que realizan su función en un punto fijo del espacio (p.ej. célula de estampación). Estos prototipos son los utilizados típicamente en las prácticas de sistemas de *Automatización Industrial* y en ellos el controlador FT está permanentemente conectado al PC donde se ejecuta el programa principal de control.

El *modo autónomo* (ver Fig. 3) permite el control del sistema físico desde el propio controlador FT sin necesidad de un PC auxiliar. En este caso el programa principal de control realizado por el usuario se descarga vía serie a la memoria interna del controlador FT donde se ejecuta. En este modo de trabajo el programa de usuario está sujeto a las restricciones impuestas por el hardware del controlador FT (sobre todo por el tamaño de la memoria) y el lenguaje de programación específico a utilizar (LLWin o ROBO Pro, según el modelo de controlador FT empleado).

El *modo autónomo* es más apropiado para prototipos dotados de movilidad capaces de desplazarse en el espacio. Éste es el caso de los robots móviles educativos en los que el programa de control se ejecuta de forma local en el controlador FT (sin conexión con un PC auxiliar).

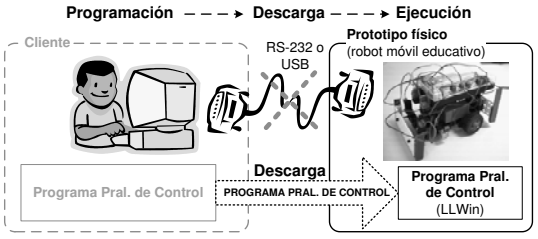

Figura 3. Esquema de trabajo en *modo autónomo*

En los laboratorios de la ESI (UCLM) se ha recurrido a prototipos físicos sencillos construidos por los estudiantes (habitualmente trabajando en grupos de dos o tres integrantes) sacrificando el realismo a cambio de la flexibilidad para proponer objetivos de aprendizaje centrados en el desarrollo de habilidades sociales, de diseño y comprensión conceptual, ya que se entiende que la construcción de maquetas complejas excede los objetivos de currículo de un ingeniero informático. Los objetivos didácticos conceptuales en los que se centra el enfoque de las prácticas de laboratorio se puede resumir en los siguientes:

- Conocer los distintos elementos de un sistema electromecánico automático y su función: *sensores, actuadores, controladores, sistemas de transmisión de potencia, sistemas de comunicaciones, sistemas de conversión de señal, etc.*
- Diferenciar los distintos tipos de sistemas automáticos atendiendo al número de estados del mismo y el mecanis-

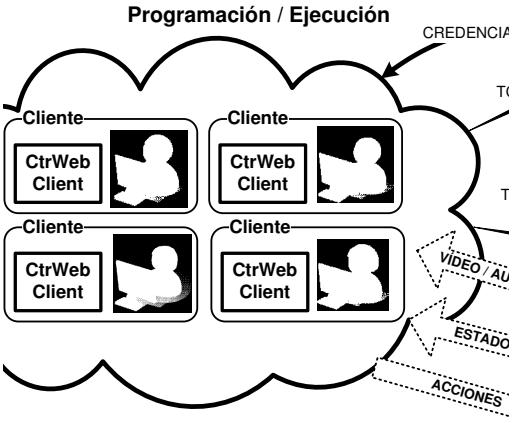

Figura 4. *Modo remoto* de funcionamiento con *CtrWeb*

mo de evolución temporal de las magnitudes físicas que los describen: *sistemas continuos, sistemas secuenciales, sistemas híbridos*.

- Conocer las estrategias de programación de los sistemas automáticos y las diferencias entre los *programas de proceso de datos* y los *programas de control* de dispositivos físicos.

Para conseguir los objetivos didácticos deseados las actividades propuestas en las prácticas de laboratorio consisten en la construcción de sistemas físicos y su programación posterior para automatizarlos. Los ejercicios propuestos son de tipo abierto ya que se indican los requisitos básicos del control quedando a criterio del alumno los aspectos de su implementación (lenguaje de programación, metodología de programación, etc.). De este modo se favorece la iniciativa y la consolidación de habilidades de diseño. Considerando los objetivos didácticos mencionados, los prototipos físicos quedan sujetos a las siguientes restricciones:

- Deben ser simples de construir. La idea es que los alumnos integrantes de cada puesto de laboratorio sean capaces de replicar el sistema propuesto en el plazo de tiempo establecido en una sesión práctica (máx. de 2 hrs.).
- Funcionalidad limitada a la capacidad hardware del controlador empleado (nº de E/S, tipo de E/S, etc.).

Durante los cursos académicos en los que se ha seguido el planteamiento descrito los alumnos han valorado muy positivamente la experiencia. Sin embargo, se entiende la necesidad de abordar actividades prácticas en sistemas físicos más complejos y realistas que los ofertados hasta la fecha. En este sentido hay empresas (Staudinger GmbH, 2009) que comercializan maquetas (compatibles con fischertechnik) de modelos físicos de gran realismo listas para su control mediante autómatas programables comerciales (PLC).

El trabajo con maquetas complejas implicaría cambios en los métodos docentes antes expuestos sugiriendo la necesidad de un mecanismo para que los estudiantes puedan compartir un prototipo físico desde todos sus puestos de trabajo e incluso hacerlo desde una ubicación remota al laboratorio. De este modo surge la necesidad de disponer de la herramienta *CtrWeb* que facilite el trabajo remoto con los sistemas físicos en estudio (Lillo, 2009). Dicha herramienta aportaría todas las ventajas atribuidas a los laboratorios remotos potenciadas aún más en un campus distribuido geográficamente como la UCLM con centros en Albacete, Almadén, Ciudad Real, Cuenca, Talavera de la Reina y Toledo.

3. ALCANCE Y OBJETIVOS DE *CTRWEB*

Vista la necesidad de una herramienta que permitiese la programación remota de sistemas físicos para su automatización se plantearon tres objetivos principales para el trabajo aquí descrito:

- Desarrollo de una API basada en una plataforma distribuida de código abierto que permita la programación y control de un prototipo físico desde una localización remota. Dicha API permitirá la programación del controlador secuencial en la parte del cliente proporcionando mecanismos transparentes de comunicación con el servidor.
- Implementación de un servidor de *streaming* capaz de ofrecer tanto vídeo como audio permitiendo la supervisión remota del prototipo bajo control desde el lado del cliente.
- Gestión de uso del prototipo físico desde un módulo (bloque de página pral.) específico creado para *Moodle*. Gracias a este módulo el acceso de los usuarios al laboratorio remoto estará integrado dentro de la plataforma institucional de campus virtual empleada en los cursos.

Con la API propuesta se añade un nuevo modo de funcionamiento, denominado *remoto*, a los ya comentados anteriormente en las Figs. 2 y 3. El *modo remoto* extiende las posibilidades del *modo on-line*, ya que ahora la ejecución del programa principal de control es distribuida entre el PC *cliente* y el PC *servidor de control* (ver Fig. 4). Esta es una gran diferencia conceptual con respecto a los trabajos analizados en la bibliografía en los que el controlador se implementa del lado del servidor.

Este nuevo modo de funcionamiento permite el trabajo simultáneo de distintos usuarios durante el proceso de programación siendo necesario solicitar el acceso exclusivo al prototipo físico para la ejecución de programas de control de la plataforma remota. El acceso remoto se puede realizar bien desde los mismos puestos de laboratorio en el que se encuentra el prototipo a controlar o bien desde un lugar distante. En este último caso el servidor de *streaming* facilita al usuario realimentación visual

y auditiva (mediante una *webcam* convencional) del comportamiento del sistema físico bajo control.

4. COMPONENTES DE CTRWEB

El sistema desarrollado para *CtrWeb* se compone de tres elementos distribuidos que comparten información mediante varios protocolos de comunicación (ver Fig. 4): 1. *cliente(s)*, 2. *servidor de control* y 3. *servidor de gestión*. A continuación se describe más en detalle cada uno de los componentes.

4.1 Clientes CtrWeb

Un *cliente CtrWeb* es un PC con conexión Ethernet (TCP/IP) en el que un estudiante programa la aplicación principal de control de un prototipo físico determinado. A través de la conexión TCP/IP el *cliente* tiene acceso tanto al *servidor de control* conectado al prototipo físico que se desea controlar, como al *servidor de gestión* del que obtiene las credenciales que garantizan el acceso al *servidor de control* durante los periodos de tiempo otorgados.

Los *clientes* ejecutan la componente cliente de la API de *CtrWeb (CtrWeb Client)*. Dicha API proporciona el acceso a distintos tipos de información disponible en el programa principal de control:

- *Credenciales de acceso* al servidor de control. Es preciso gestionar los accesos al control de un prototipo único compartido por un número de clientes simultáneos que puede abarcar el conjunto de estudiantes matriculados en el curso de laboratorio (habitualmente entre 30 y 40 estudiantes).
- *Acciones de control*. La aplicación de control en el cliente envía acciones para que las ejecute el actuador correspondiente del prototipo remoto.
- *Estado del prototipo*. La aplicación puede acceder a los valores obtenidos por los sensores instalados en el prototipo remoto y así al estado instantáneo de éste.
- *Realimentación visual y sonora* procedente de una cámara web instalada en el *servidor de control*. De este modo se puede tener información realista para la supervisión del estado del prototipo.

4.2 Servidor de control

El *servidor de control* es único y actúa como enlace entre los *clientes* y el prototipo físico al que está conectado mediante una conexión serie (RS232 o USB dependiendo del modelo concreto de controlador FT empleado). El *servidor de control* ejecuta la parte servidora de la API desarrollada *(CtrWeb Server)* que actúa como un *middleware* entre la aplicación de control desarrollada por los clientes y la API *JavaFish* (Müller, 2008) que establece la conexión con el prototipo físico a través de la comunicación con *FT server* (ubicado en el controlador FT). *CtrWeb Server* añade funcionalidad adicional para la retransmisión (hacia los clientes) de *streaming* de vídeo y audio obtenido desde una *webcam* instalada junto al prototipo físico. Dicha información debería permitir la supervisión de la ejecución del programa por parte de los clientes.

El *servidor de control* también establece conexión con el *servidor de gestión* para habilitar la conexión del prototipo a los *clientes* autorizados y así arbitrar el uso compartido de recursos por todos los estudiantes del curso.

4.3 Servidor de gestión

El *servidor de gestión* es un servidor corporativo que proporciona las credenciales de acceso a los clientes remotos para el trabajo con el prototipo físico compartido. La gestión de accesos a través del *servidor de gestión* se realiza mediante el LMS *Moodle* (Moodle Org, 2009) para el que se ha desarrollado el *CtrWeb Moodle Module (CtrWeb M-Module)* como bloque de página principal para el curso en el que se contempla el uso de *CtrWeb*. De este modo sólo los alumnos matriculados en el curso tienen acceso al control del prototipo en los tiempos que el servidor les otorgue.

5. TECNOLOGÍAS APLICADAS

La Figura 5 muestra las distintas tecnologías empleadas en los actores implicados en *CtrWeb*, siendo importante destacar que todas las implementaciones utilizadas son de código abierto. Entre todas ellas quizás la más importante sea la que hace posible la comunicación entre el *cliente* y el *servidor de control*. Existen múltiples posibilidades para la implementación del mecanismo de comunicaciones mencionado: CORBA, Java RMI *(Remote Method Invocation)*, DCOM, DCE/RPC, etc. En *CtrWeb* se ha decidido el empleo de Java RMI (Sun Microsystems, Inc., 2004*a*) ya que el lenguaje Java se empleará como base de desarrollo de las aplicaciones cliente. A parte de la disponibilidad de RMI, como principales ventajas del uso de Java se pueden mencionar las siguientes:

- Es un lenguaje multiplataforma que proporciona una gran generalidad al desarrollo.
- Curva de aprendizaje mínima en el caso de los alumnos a los que está destinada la herramienta *CtrWeb* ya que Java es el lenguaje de programación al que están habituados en las materias cursadas durante la titulación.
- Ofrece la posibilidad de reutilización de componentes debidos a terceros entre los que se encuentran bibliotecas para la conexión con diferentes sistemas gestores de bases de datos.
- Soporta el paradigma de programación orientado a objetos OOP *(Object Oriented Programming)* con soporte para herencia, programación multihilo, sobrecarga, etc. que lo convierten en un lenguaje de programación moderno muy completo.
- Existen herramientas que soportan todo el ciclo de vida del software Java lo que facilita enormemente el análisis, diseño, implementación y pruebas del software.

En el *servidor de control* existe un conjunto de herramientas que describimos brevemente a continuación (de abajo a arriba en la Fig. 5):

- *JavaFish*. Es una biblioteca Java realizada por Ulrich Müller (Müller, 2008) y con la que es posible realizar la programación de los controladores FT en *modo on-line*. Básicamente son un conjunto de funciones que encapsulan el protocolo de comunicación serie entre una aplicación cliente y el servidor *FT Server* ejecutado como parte del *firmware* de los controladores FT.
- JMF (Sun Microsystems, Inc., 2004*b*). Es una biblioteca que habilita la manipulación de audio y vídeo en las aplicaciones basadas en Java. Esta biblioteca se emplea para el manejo del *streaming* de vídeo y audio hacia los clientes de *CtrWeb*.

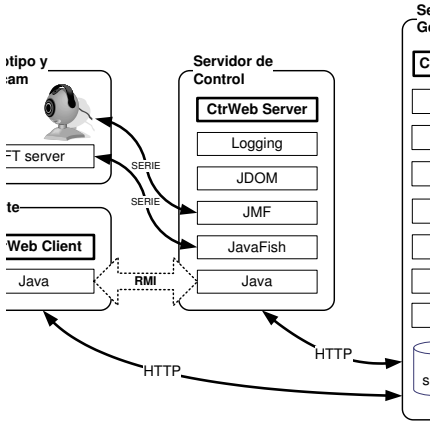

Figura 5. Principales tecnologías empleadas en *CtrWeb*

- JDOM. Es un *modelo de objeto documento* (DOM) específico para su integración con Java. JDOM se emplea en *CtrWeb* como herramienta para configurar las comunicaciones entre los clientes y los servidores de control y gestión.
- Logging. Es una API para facilitar los registros de ejecución *(logs)* de una aplicación Java.

El acceso a *Moodle* es proporcionado por el servidor web *Apache* en el que son precisos algunos módulos específicos para la programación mediante PHP, el acceso a bases de datos (MySQL y SQLite) y la gestión del correo electrónico dirigido a los usuarios de la aplicación.

La elección de *Moodle* se debe a que es un LMS empleado en un número creciente de universidades que ha ido relegando a la plataforma *WebCT* siendo hoy en día sin lugar a dudas la plataforma más utilizada en la universidad española. Según datos del estudio realizado por profesores de la Universidad de Oviedo (Álvarez *et al.*, 2008) *Moodle* tiene una cuota de uso del 34.55 % frente al 27.27 % de WebCT/Blackboard, con una tendencia favorable al primero. Entre las ventajas de *Moodle* debe señalarse que es una herramienta de código abierto, gratuita y de libre distribución ampliable y con posibilidades de fácil adaptación a unas necesidades concretas. En concreto en la UCLM (desde el curso 2006/07) es la herramienta que proporciona soporte al *Campus Virtual* (Univ. Castilla-La Mancha, 2010) y dentro de éste a las materias impartidas en la Escuela Superior de Informática (ESI) desde el área de Ingeniería de Sistemas y Automática (ISA).

Uno de los problemas habituales en los laboratorios remotos es el acceso a determinados puertos TCP/UDP limitados por los *firewalls* interpuestos. Aunque *CtrWeb* emplea tanto puertos TCP (peticiones RMI) como UDP (servicio de *streaming*), el acceso al laboratorio remoto empleando la *red privada virtual* de la UCLM (VPN) evita cualquier tipo de problema de acceso filtrado por el *firewall*. No obstante, las conexiones encargadas de transmitir el audio y vídeo a través de *streaming* pueden superar los *firewall* interpuestos utilizando el protocolo RTSP que utiliza el puerto 554 para el tráfico UDP. El tráfico TCP también podría superar los *firewall* empleando la capacidad de RMI para utilizar un mecanismo de invocación encapsulado bajo protocolo HTTP (confiable para los *firewall*). No obstante, la API suministrada con *CtrWeb* habilita la transparencia del proceso de comunicación entre cliente y servidor requiriendo únicamente la edición de un fichero de configuración en el que se indican los puertos empleados para las conexiones TCP y UDP.

6. ARQUITECTURA SOFTWARE DE CTRWEB

Para el diseño de la herramienta *CtrWeb* se ha elegido una *arquitectura multicapa* (Rumbaugh *et al.*, 2004). Este tipo de arquitectura se basa en que cada capa proporciona servicios a los niveles superiores reduciéndose el acoplamiento del sistema ya que cada capa sólo conoce a su adyacente. El sistema multicapa ofrece numerosas ventajas que se pueden resumir en los siguientes puntos:

- El diseño emplea diferentes niveles de abstracción que permiten abordar un problema complejo por descomposición en subproblemas más simples.
- Los componentes desarrollados son más fácilmente reutilizables.
- Se puede optimizar el rendimiento de algunos componentes sin afectar por ello a otros niveles.
- El sistema final es fácilmente escalable pues es sencillo modificar componentes de forma aislada.
- Es posible la distribución de los componentes entre distintas unidades de procesamiento.

La Fig. 6 muestra la relación existente entre los distintos componentes del sistema global.

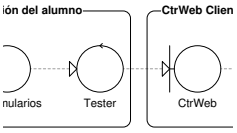

Figura 6. Arquitectura software de *CtrWeb*

El empleo de *CtrWeb* es sencillo pero requiere conocimientos previos de programación con Java. Los usuarios (alumnos) programarán sus aplicaciones de control proporcionando las funcionalidades que consideren oportunas pero a su vez emplearán la capacidad de conexión con el servidor de control al que se encuentra conectado el prototipo físico a controlar.

La programación con *CtrWeb* representa un mínimo esfuerzo adicional para el alumno frente al uso de bibliotecas habituales empleadas en Java. Su uso puede resumirse como sigue. En primer lugar se instancia un objeto que encapsula a la parte cliente mediante el constructor:

> *CtrWeb*(String ip, String portUDP, String user, String passw);

Una vez creada la instancia ya es posible controlar el acceso al prototipo físico utilizando los métodos ligados a dicha instancia:

> public int *getInputAnalogic*(int inputNr);
> public boolean *getInputDigital*(int inputNr);
> public int *setOutput*(int motorNr, int action);

Una funcionalidad importante que se proporciona es el empleo de *streaming* de vídeo y audio para poder realizar control supervisor. En este caso es preciso invocar el método siguiente:

> public Reproductor *getReproductor*();

De este modo se crea un objeto *Reproductor*, siendo posible reproducir secuencias de vídeo y audio mediante el método: public void *play*();

Finalmente para visualizar el vídeo hay que añadir el componente visual del objeto *Reproductor* al panel de la interfaz de usuario del programa cliente como se muestra en la siguiente porción de código:

```
robot=new CtrWeb(ip, port, UDPport, user, passw);
player=this.robot.getReproductor();
player.play();
Component comp=player.getVisualComponent();
comp.resize(320, 240);
this.jPanel().add(comp);
```

Como *CtrWeb Client* es una biblioteca, en su diseño se elimina la capa de presentación quedando únicamente la capa de comunicaciones que asume toda la funcionalidad de la parte cliente. Esta capa de comunicaciones se apoya en otra capa que gestiona las funciones involucradas en la reproducción de datos multimedia.

CtrWeb Server emplea a su vez un diseño multicapa para procesar las acciones recibidas destinadas a su ejecución sobre el prototipo físico (ver Fig. 6).

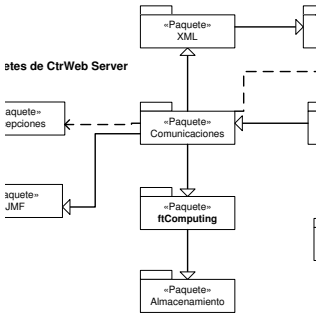

Figura 7. Diagrama de paquetes de *CtrWeb Server* y *CtrWeb Client*

En el diseño de *CtrWeb Server* se ha sustituido la capa de presentación por una de comunicaciones apareciendo nuevas capas (ver Fig. 7) que pasamos a describir brevemente:

- *Capa de dominio*. Inicializa el servidor RMI encapsulando las clases de control y acceso al sistema. Adicionalmente alberga una clase contenedora *Alumno* que representa al usuario conectado al sistema.
- *Capa de comunicaciones*. Alberga las clases encargadas de la inicialización del servidor de control del prototipo físico y del servidor multimedia que retransmite vídeo y audio para supervisión.
- *Capa XML*. Hospeda la clase que procesa el fichero XML de configuración de la conexión con los servidores de control y gestión (dirección IP, puertos, usuario, clave, etc.) para crear un objeto en memoria que pueda ser consultado durante la ejecución del programa de usuario.
- *Capa Config*. Permite el inicio del sistema con propiedades configuradas dinámicamente.
- *Capa JMF*. Encapsula la funcionalidad de captura de señales procedentes de una fuente de vídeo y audio para su publicación mediante *streaming*.
- *Capa FtComputing*. Proporciona la comunicación con el prototipo físico.
- *Capa de almacenamiento*. Se encarga de la comunicación con el sistema gestor de base de datos (SGBD).
- *Capa de excepciones*. Gestiona los errores que se producen en el sistema.

6.1 Gestión de la persistencia y de excepciones

La herramienta *CtrWeb* debe manipular y almacenar múltiples datos durante su ejecución. Para la gestión y almacenamiento de dichos datos se han empleado dos SGBD: MySQL y SQLite. Todos los datos relativos al uso del sistema se almacenan en la base de datos MySQL para su posterior tratamiento (p.ej. estadísticas de uso, gestión de horarios, etc.). Para gestionar la cola FIFO (*First In, First Out*) de acceso al sistema físico compartido se ha elegido SQLite ya que dicha cola debe consultarse y modificarse muy frecuentemente y debe satisfacer altos requerimientos de eficiencia en consultas y modificación. Para asegurar el control total de la base de datos se hace uso del patrón *singleton* implementado en Java.

La gestión de las excepciones y errores que se producen durante la ejecución de *CtrWeb* se encapsula mediante la clase *ExceptionCtrWeb*, ya que es preciso disponer de un mecanismo que permita la gestión de errores para su tratamiento dentro de la aplicación cliente del usuario final.

6.2 Integración de CtrWeb y Moodle

La integración de *CtrWeb* y *Moodle* tiene lugar mediante el *CtrWeb M-Module* programado con PHP. En la Fig. 8 se observa el aspecto del nuevo módulo añadido a la plantilla institucional de la UCLM para *Moodle*. Dicho módulo se ha implementado como bloque principal del curso en el que se va a utilizar el laboratorio remoto. La funcionalidad que aporta la integración de *CtrWeb* con *Moodle* puede resumirse como sigue:

- El acceso al prototipo físico sólo es posible para los alumnos matriculados en el curso o aquellos autorizados por el administrador/es (profesor/es) del curso.
- Gestión de uso mediante una cola FIFO. Una vez solicitado el uso, la petición se encola hasta que se asigna el turno. El tiempo máximo de uso es configurable y también el tiempo permitido con actividad nula. De este modo se garantiza el aprovechamiento de la infraestructura por parte de todos los usuarios. Las solicitudes también pueden anularse para otorgar el turno a otros usuarios.
- Monitorización del uso instantáneo del prototipo.
- Acceso a la documentación de las bibliotecas y paquetes de la herramienta *(JavaDoc)*.
- Acceso a través del árbol del repositorio *Subversion* tanto al código de desarrollo de la herramienta (para posibles modificaciones y ampliaciones) como a la versión final de la herramienta.

La configuración del *CtrWeb M-Module* es bastante sencilla y muy fácil de realizar en la UCLM (u otras instituciones). En primer lugar el código fuente (PHP) del módulo debe copiarse en el directorio correspondiente a los bloques de la instalación *Moodle* en el servidor. Dicho directorio es habitualmente (en sistemas Linux): > */var/www/moddlev7/blocks*. Una vez copiado el código en la carpeta apropiada, el módulo pasa a ser visible en la instalación local. El administrador del *servidor de gestión* puede activar el *CtrWeb M-Module* simplemente pulsando sobre la opción *Modules/Blocks* indicando los cursos en los que debe estar visible el nuevo bloque *CtrWeb*. Para el

Figura 8. *Moodle* y módulo *CrtWeb*

correcto funcionamiento de dicho bloque el *servidor de gestión* debe tener instalado también el software que se indica a continuación:

- SQLite (en Linux mediante paquete *sqlite3*). Se emplea en la gestión de la cola de espera para el acceso de usuarios al prototipo físico. La creación de la base de datos la realiza automáticamente el bloque *CtrWeb*.
- Servicio de envío de correo *(SMTP, Simple Mail Transfer Protocol)* que emplearán *Moodle* y el bloque *CtrWeb* para el envío de correo a los usuarios.

El módulo implementado se ha concebido como *bloque principal* disponible para el curso y no cómo *recurso* o *actividad* del curso. Esta elección atiende a la conveniencia en el contexto de trabajo de los autores con la justificaciones que a continuación se indican:

- Los *recursos* están disponibles para todos los cursos de la institución independientemente de la orientación de éstos (humanidades, ciencias, ingenierías, etc.) y son los administradores del curso (profesores) los que deciden su inclusión en el curso. A los autores no les pareció conveniente hacer disponible a toda la universidad un recurso que sólo va a ser empleado en cursos relacionados con el área de ISA.
- No es preciso diseñar un nuevo tipo de actividad para *Moodle* que contemple el uso de un laboratorio remoto. Las actividades existentes ya permiten tratar con un escenario de trabajo basado en un laboratorio remoto (p.ej. realización de un programa de control y entrega de los ficheros a través de tareas que impliquen subida de archivos).

7. DISCUSIÓN Y CONCLUSIONES

En este trabajo se presenta una herramienta de código abierto *(CtrWeb)* que permite la programación distribuida de aplicaciones de control remoto secuencial de prototipos físicos (fischertechnik) empleados en laboratorios docentes de automatización. Las principales características de dicha herramienta se pueden resumir en los siguientes aspectos:

- Acceso a los dispositivos físicos mediante conexión TCP/IP a un *servidor de control* donde se encuentra conectado el prototipo físico a través de enlace serie.
- Disponibilidad, en la parte cliente, de mecanismos de supervisión de la instalación bajo estudio a través tanto de retransmisión de vídeo y audio, como de instrucciones de parada y detección de situaciones de emergencia.
- Gestión de usuarios y conexiones mediante integración de la herramienta con un LMS abierto *(Moodle)* garantizando así el acceso sólo a los usuarios autorizados durante el tiempo planificado.
- Arquitectura de comunicaciones basada en Java RMI que habilita la programación del controlador en la parte cliente y la invocación de métodos remotos para llevar a cabo las operaciones de E/S en el *servidor de control*.
- Arquitectura software, independiente del hardware, en la que se emplean las técnicas de ingeniería de software más destacadas (OOP, diseño multicapa, etc.).
- Implementación completa en código abierto lo que garantiza las posibilidades de modificación, adaptación y en definitiva la portabilidad a diferentes contextos de implantación.

Los usuarios finales de la herramienta desarrollan su aplicación con una API que proporciona la funcionalidad necesaria para el control de prototipos físicos y supervisión mediante *streaming* de vídeo y audio (en la Fig. 9 se muestra una aplicación de ejemplo para un test de control de la estación de estampación).

7.1 Experiencia de uso

CtrWeb es un trabajo cuya primera versión ha visto la luz a principios del curso académico 2009/2010 (finales de septiembre 2009). Por este motivo durante dicho curso se ha ofrecido

Figura 9. Ventanas de ejecución de sistema construido con *CtrWeb*

la herramienta para que aquellos alumnos que lo deseen puedan usarla de forma opcional y así poder evaluar sus fortalezas y debilidades. La metodología para evaluar la experiencia de uso ha sido la elaboración de un cuestionario con cinco preguntas. Las primeras tres preguntas son de respuesta cerrada y en las dos finales se pide la opinión personal del alumno. Dicha encuesta se ha realizado con ayuda de *Moodle* para que el alumno la pueda contestar de forma individual y según su preferencia. Las preguntas son las siguientes:

1. (P1) ¿Te parece interesante la propuesta de un laboratorio remoto mediante la herramienta *CtrWeb*? (Opciones: a) totalmente de acuerdo, b) de acuerdo, c) indiferente, d) en desacuerdo y e) totalmente en desacuerdo)
2. (P2) ¿Estarías dispuesto a emplear *CtrWeb* para la realización de prácticas durante el curso? (Opciones: Ídem pregunta anterior)
3. (P3) ¿Qué tipo de laboratorio prefiere? (Opciones: a) remoto, b) híbrido, c) convencional y d) indiferente)
4. ¿Cuáles son en tu opinión las fortalezas y debilidades de un laboratorio remoto frente a uno convencional?
5. ¿Cuáles son en tu opinión las características que más valoras en la herramienta para laboratorios remotos?

A continuación se comenta los resultados obtenidos que aparecen resumidos en la Fig. 10.

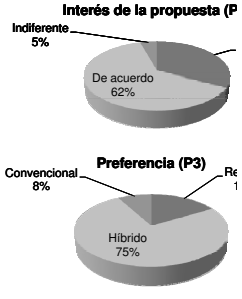

Figura 10. Resultados estadísticos de la experiencia de uso

Como el empleo de la herramienta es opcional la participación de los alumnos es relativamente baja pues se sitúa en torno al 69 % (24/35) frente a los que habitualmente participan de las actividades propuestas en el curso (el total de alumnos matriculados en el curso es 38). Los alumnos demuestran un alto grado de interés en la propuesta y predisposición a su uso. Sin embargo es claro el amplio grado de «reserva» puesto de manifiesto en el alto porcentaje de alumnos que se manifiestan de acuerdo pero no en su grado máximo.

Sobre la preferencia de uso los alumnos se manifiestan de forma mayoritaria por una modalidad híbrida de laboratorio (75 %) que combine tanto el método tradicional como la posibilidad de operación remota, preferida por el 17 %. Hay un porcentaje mínimo de alumnos (8 %) que no desea en ningún modo enfrentarse a la novedad que supone los laboratorios remotos.

En las respuestas a las cuestiones 4 y 5 existe consenso en señalar la disponibilidad horaria y geográfica como las principales fortalezas del laboratorio remoto. A parte de estos aspectos también se mencionan como fortaleza la cantidad y calidad de la documentación ofrecida (p.ej. ejemplos, video tutoriales, etc.).

Respecto de las debilidades, los alumnos mencionan su interés en que se mejore el soporte multilenguaje y multiplataforma, así como la inclusión de un simulador que permita la ejecución *of ine* de los programas realizados.

A pesar de la valoración positiva de la herramienta y en general de las prácticas de laboratorio como elemento motivador y facilitador del aprendizaje, los alumnos manifiestan dudas y reticencias relativas al uso de un laboratorio remoto. Las principales reticencias de los alumnos relativas al uso de un laboratorio remoto están relacionadas con los siguientes aspectos:

- *Asistencia técnica*. El trabajo en modo remoto limita las posibilidades de asistencia por parte del profesor y/o otros compañeros (p.ej. resolviendo dudas, aprendiendo de las soluciones propuestas por otros, etc.).
- *Fiabilidad*. Los alumnos muestran su preocupación por la posibilidad de que se produzcan fallos (p.ej. sobrecarga

en servidores, fallos mecánicos o de conexionado en los prototipos, fallos en las líneas eléctricas, etc.) que le impidan alcanzar los objetivos marcados en los plazos propuestos.
- *Tiempo de respuesta*. Dependiendo del tráfico y características del acceso a red, el tiempo de respuesta que se obtiene es variable ya que no se dispone de mecanismos que garanticen un ancho de banda mínimo para asegurar que el sistema cumple unos requisitos de tiempo de respuesta mínimos.
- *Realimentación visual*. La información procedente de la cámara web no siempre es de suficiente calidad como para resultar útil. En la actualidad no existe ningún sistema de iluminación controlada que garantice una calidad homogénea de imagen las 24 horas del día.

Los mismos alumnos ofrecen algunas de las estrategias para vencer las iniciales reticencias que existen en el empleo de laboratorios remotos. Es fundamental informar al alumno de los objetivos y procedimientos diseñando cuidadosamente unas prácticas cuya metodología evite en lo posible las limitaciones que más arriba se mencionan. algunas estrategias mencionadas y que conviene estudiar son: a) hacer optativas las prácticas en remoto, b) restringir las franjas horarias de uso del laboratorio al horario lectivo, c) asegurar la presencia de un instructor en el laboratorio durante el horario de posibles conexiones, d) añadir recurso de comunicación *online* con el instructor (p.e. chat), etc.

7.2 Limitaciones y posibles mejoras

Debe señalarse que la herramienta presentada ofrece algunas limitaciones que serán objeto de futuras revisiones de desarrollo:

1. Las estrategias de control para las que está pensada la herramienta se limitan al *control dirigido por eventos (control secuencial)*. En este sentido debería contemplarse el empleo de controladores de mayores prestaciones y el análisis en profundidad de los sistemas de control en red a emplear *(networked control systems)*.
2. Una de las limitaciones apuntadas por los estudiantes encontradas en la programación de los prototipos empleados en las prácticas es la necesidad de un simulador de E/S que permita la depuración de los programas previamente a su ejecución sobre la plataforma física. Esta limitación incluso aparece en los laboratorios convencionales ya que en ocasiones los estudiantes desearían probar sus programas en modo *off-line*. El proyecto *FTPack* (Fernández, 2010) está destinado a proveer de un entorno de desarrollo completo, incluido el simulador mencionado, sobre Eclipse para Java, C# y Python. En la actualidad se trabaja en la integración de *CtrWeb* sobre dicho entorno.
3. El desarrollo de los programas en la parte cliente están limitados al lenguaje de programación Java debido en parte al acoplamiento existente entre dicho lenguaje y el mecanismo de comunicación RMI. No obstante es posible obviar esta limitación recurriendo a la programación de un «envoltorio» de Java RMI en otros lenguajes finales. En esta dirección se ha trabajado en el proyecto *EduPLC* (Franco, 2009) cuyo objetivo es proporcionar un entorno de programación para lenguajes estandarizados bajo IEC-1131-3 especialmente dirigidos a la automatización industrial.
4. La herramienta está pensada asumiendo que existe un único prototipo físico conectado al *servidor de control*. Para superar esta limitación sería preciso modificar tanto el *servidor de gestión* como el *servidor de control*. En el primero habría que gestionar las conexiones de los usuarios a los distintos prototipos y el segundo debería arbitrar el acceso a las distintas plataformas físicas conectadas.
5. Necesidad de garantizar la conexión con el *servidor de gestión*. La conexión con dicho servidor proporciona las credenciales de acceso al sistema pero si una vez obtenidas éstas la conexión falla, el usuario perdería la capacidad de control de la plataforma. No obstante, *FT server* proporciona mecanismos de detención de actuadores en caso de pérdida de conexión por un tiempo superior a 300 ms.
6. Las comunicaciones entre cliente y servidores no están cifradas y por tanto son susceptibles ante ataques. Es preciso mejorar los aspectos relacionados con la seguridad en las comunicaciones pues es un aspecto en el que no se ha profundizado.

A pesar de las limitaciones comentadas, *CtrWeb* ofrece la posibilidad de realizar la programación remota de un prototipo físico real con un planteamiento abierto en el sentido de que las funcionalidades que se pueden implementar son prácticamente ilimitadas. Además al haberse realizado en Java se permite el uso de todos los recursos disponibles para dicha plataforma software.

Con la herramienta presentada en este trabajo es posible abordar nuevas estrategias de enseñanza a distancia en el laboratorio de automatización ya que es posible compartir los recursos físicos entre todos los alumnos de un curso y abordar ejercicios con instalaciones más complejas a las habitualmente empleadas. Dichas estrategias ofrecen grandes ventajas en el caso de universidades, como la UCLM, donde los centros educativos presentan gran dispersión geográfica. Por este motivo se ha integrado la herramienta desarrollada en el Campus Virtual de la UCLM mediante un bloque nuevo (*CtrWeb M-Module*) disponible en la instalación local de *Moodle*.

AGRADECIMIENTOS

Los autores desean expresar su agradecimiento al Depto. de Ing. Eléctrica, Electrónica, Automática y Comunicaciones (IEEAC) de la Universidad de Castilla-La Mancha (UCLM) y a la Escuela Superior de Informática (ESI) por haber proporcionado las infraestructuras y el soporte técnico para la realización de este trabajo.

REFERENCIAS

Aktan, B., C. A. Bohus et al. (1996). Distance learning applied to control engineering laboratories. *IEEE Transactions on Education* **39**(3), 320–326.

Aliane, N. (2008). Limitaciones pedagógicas de los laboratorios remotos de control. In: *Actas de las XXIX Jornadas de Automática*. Tarragona, España.

Aliane, N., A. Martínez et al. (2007). Labnet: A remote control engineering laboratory. International Journal of Online Engineering, 3 (2)
http://www.i-joe.org/ (último acceso nov. 2009).

Álvarez, V. M., M. del Puerto, J. R. Pérez and I. Gutiérrez (2008). Presente y futuro del desarrollo de plataformas Web de elearning en educación superior. In: *V Simposio Pluridisciplinar sobre Diseño y Evaluación de Contenidos Educativos Reutilizables*. Salamanca.

Candelas, F. A. and J. Sánchez (2005). Recursos didácticos basados en internet para el apoyo a la enseñanza de materias del área de ingeniería de sistemas y automática. *Revista Iberoamericana de Automática e Informática Industrial* **2**(2), 93–101.

Casini, M., D. Prattichizzo and A. Vicino (2003). The automatic control telelab: A user-friendly interface for distance learning. *IEEE Transactions on Education* **46**(2), 252–257.

Costa, R., M. Vallés, L.M. Jiménez, L. Díaz, A. Valera and R. Puerto (2010). Integración de dispositivos físicos en un laboratorio remoto de control mediante diferentes plataformas: Labview, Matlab y C/C++. *Revista Iberoamericana de Automática e Informática Industrial* **7**(1), 23–34.

Cruz de la, F., M. Díaz, S. Zerpa and D. Giménez (2010). WebLABAI: Laboratorio remoto de Automatización Industrial. *Revista Iberoamericana de Automática e Informática Industrial* **7**(1), 101–106.

Domínguez, M., P. Reguera and J. J. Fuertes (2005). Laboratorio remoto para la enseñanza de la automática en la universidad de León. *Revista Iberoamericana de Automática e Informática Industrial* **2**(2), 36–45.

Dormido, R., H. Vargas, N. Duro, J. Sánchez, S. Dormido-Canto, G. Farias, F. Esquembre and S. Dormido (2008). Development od a web-based control laboratory for automation technicians: the three-tank system. *IEEE Transactions on Education* **51**(1), 35–44.

Dormido, S. (2004). Control learning: present and future. *Annual Reviews in Control* **28**, 115–136.

Dormido, S., J. Sánchez and F. Morilla (2000). Laboratorios virtuales y remotos para la práctica a distancia de la automática. In: *Actas de las XXI Jornadas de Automática*. Sevilla.

Fernández, Mª del Milagro (2010). FTPack: Framework de programación de sistemas educativos basado en Eclipse. Proyecto final de carrera. Escuela Superior de Informática, Univ. de Castilla-La Mancha. Ciudad Real (España).

fischertechnik (2010). Building blocks for life. http://www.fischertechnik.de/en/ (último acceso abr. 2010).

Franco, Alfonso (2009). EduPLC: entorno de programación para interfaces fischertechnik. Proyecto final de carrera. Escuela Superior de Informática, Univ. de Castilla-La Mancha. Ciudad Real (España).

García, M. and R. Rallo (2005). Towards the integration of remote laboratories into learning management systems. In: *Proceedings of the 10th IEEE Int. Conf. on Emerging Technologies and Factory Automation*.

Gasa, D., I. Garrido, R. Costa and L. Basañez (2005). Plataforma de ensayo remoto de controladores basados en autómatas finitos. *Revista Iberoamericana de Automática e Informática Industrial* **2**(2), 55–63.

GENIA - Univ. Oviedo (2010). Grupo de Entornos Integrados de Automatización. http://isa.uniovi.es/genia/ (último acceso abr. 2010).

Gillet, D., A. Vu Nguyen Ngoc and Y. Rekik (2005). Collaborative web-based experimentation in flexible engineering education. *IEEE Transactions on Education* **48**(1), 696–704.

Grupo de Educación en Automática, CEA-IFAC (2009). Laboratorios virtuales y/o remotos. http://www.cea-ifac.es/w3grupos/educontrol?q=node/33 (último acceso abr. 2010).

Guzmán, J.L., M. Domínguez, M. Berenguel, J.J. Fuertes, F. Rodríguez and P. Reguera (2010). Entornos de experimentación para la enseñanza de conceptos básicos de modelado y control. *Revista Iberoamericana de Automática e Informática Industrial* **7**(1), 10–22.

Lillo, A. (2009). CtrWeb: Herramienta de programación para telecontrol de sistemas físicos. Proyecto final de carrera. Escuela Superior de Informática, Univ. de Castilla-La Mancha. Ciudad Real (España).

Ma, J. and J. V. Nickerson (2006). Hands-on, simulated, and remote laboratories: A comparative literature review. *ACM Computing Surveys* **38**, article 7.

Macías, M. E. and I. Méndez (2008). Telelab: remote automations lab in real time. In: *Proceedings of the 38th. ASEE/IEEE Frontiers in Education Conference*.

Müller, U. (2008). Java corner - ft computing. http://www.ftcomputing.de (último acceso nov. 2009).

Moodle Org (2009). Moodle docs. http://docs.moodle.org (último acceso nov. 2009).

Rodríguez, J.A and O. Neira (2001). webPLC: una herramienta para las clases de automatización industrial. In: *Actas de las XXII Jornadas de Automática*. Barcelona, España.

Rumbaugh, James, Ivar Jacobson and Grady Booch (2004). *The Unified Modeling Language Reference Manual*. 2nd. ed.. Addison-Wesley Professional.

Salido, J. (2006). Programación on-line de la intelligent interface (ref. 30402) de fischertechnik. Technical report. Universidad de Castilla-La Mancha.

Salido, Jesús (2009). *Cibernética aplicada. Robots Educativos*. 1ª ed. ed.. Ra-Ma.

Sancristobal, E., S. Martin et al. (2008). Integration of Internet based labs and open source LMS. In: *Proceedings of the 3rd. Int. Conf. on Internet and Web Applications and Services* (IEEE Computer Society, Ed.). pp. 217–222.

Staudinger GmbH (2009). Product overview simulation models. http://www.staudinger-est.de/ (último acceso nov. 2009).

Sun Microsystems, Inc. (2004*a*). Java remote method invocation (Java RMI). http://java.sun.com/j2se/1.5.0/docs/guide/rmi/index.html (último acceso nov. 2009).

Sun Microsystems, Inc. (2004*b*). Java se desktop technologies (JMF). http://java.sun.com/javase/technologies/desktop/media/jmf/ (último acceso nov. 2009).

Univ. Castilla-La Mancha (2010). Campus virtual de la uclm. https://campusvirtual.uclm.es.

Valera, A., M. Vallés and J. L. Díez (2005). Simulación y control de procesos físicos de forma remota. *Revista Iberoamericana de Automática e Informática Industrial* **2**(2), 20–29.

Zuluaga, C. A., C. G. Sánchez and E. A. Rodríguez (2005). Laboratorio de automática vía internet (lavi). *Revista Iberoamericana de Automática e Informática Industrial* **2**(2), 30–35.