Revista Iberoamericana de Automática e Informática Industrial

CEA comité español de automática

ISSN: 1697-7912. Vol. 7, Núm. 1, Enero 2010, pp. 54-63

http://www.revista-riai.org

BENDER 3.0, una plataforma robótica remota para aplicaciones docentes: Aplicación a Programación Concurrente

Nieves Pavón*, Joaquín Ferruz**

**Departamento de Tecnologías de la Información, Universidad de Huelva, E.P.S. de la Rábida. Carretera Huelva - La Rábida - 21071 Palos de la Frontera (Huelva), España, (e-mail: npavon@dti.uhu.es)*

***Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Escuela de Ingenieros, Camino de los Descubrimientos, s/n. Isla de la Cartuja, 41092 Sevilla, España, (e-mail: ferruz@cartuja.us.es)*

Resumen: *BENDER* 3.0 es un pequeño robot móvil usado en la asignatura Programación Concurrente. El artículo se centra en la descripción de dicho robot y en el análisis de la metodología seguida para desarrollar las prácticas de laboratorio. Se revisan las estrategias que facilitan su uso y se describe el marco de simulación que posibilita el trabajo del estudiante en casa, independientemente de si dispone o no de conexión a Internet. Finalmente, se presenta un estudio detallado de los resultados académicos obtenidos por los alumnos y se menciona el uso del robot móvil en otras asignaturas y proyectos fin de carrera. Copyright © 2010 CEA.

Palabras Clave: Programación Concurrente, Educación en Automática, Sistemas de Tiempo Real, Robótica.

1. INTRODUCCIÓN

Los importantes avances en el ámbito de la Robótica y de la Automática han comenzado a llegar a los hogares en forma de dispositivos domésticos que facilitan múltiples tareas: desde servicios de vigilancia, hasta el control biométrico de enfermos, pasando por la realización de tareas del hogar (Gates, 2007).

Las numerosas mejoras en el diseño microelectrónico y el abaratamiento de los costes de producción de ciertos robots y dispositivos automáticos han permitido que se produzca una nueva revolución donde los sistemas informáticos, electrónicos, domóticos y de control inteligente ya forman parte del paisaje, dando lugar a un nuevo concepto: la robótica de servicio con capacidades cognitivas (Aracil, Balaguer y Armada, 2008).

Este concepto abre grandes expectativas de negocio donde deberán trabajar, de forma integrada, profesionales de ramas muy diversas: Ingenieros en Informática, en Electrónica, en Telecomunicaciones, Industriales, expertos en biotecnología, en procesamiento de lenguaje natural o psicología, entre otros. Todos estos profesionales habrán de cooperar para diseñar sistemas que trabajen en entornos distribuidos, con capacidades y habilidades cognitivas, capaces de interaccionar apropiadamente con el entorno y con los seres humanos (Casals, 2007).

Por otra parte, los futuros egresados se enfrentan a profundos cambios en el entorno universitario derivados de la implantación de los nuevos planes de estudio y de un sistema de valoración de su trabajo diferente al utilizado hasta ahora: el *plan Bolonia* y el *sistema de transferencia de créditos europeos ECTS* (M.E.C., 2003), (Pagani, 2002).

Los futuros graduados deberán adquirir una serie de competencias que les permitan integrarse en los equipos de trabajo mencionados de una manera eficaz.

Para adquirir dichas competencias es necesario modificar el modo de impartición de las asignaturas que componen las actuales titulaciones y los futuros grados. Se deben buscar estrategias que permitan evaluar el trabajo de cada estudiante de un modo personalizado teniendo en cuenta sus aptitudes, sus necesidades y la coyuntura de cambio actual.

Los futuros ingenieros formarán parte de equipos multidisciplinares donde tendrán que poner a prueba, no sólo su capacidad para desarrollar los trabajos para los que han adquiridos competencias específicas sino también para interaccionar con expertos de otras ramas y, en último lugar, con el cliente o usuario final.

Para conseguir este objetivo fundamental es vital aprovechar las nuevas tecnologías con el fin de mejorar el concepto de "Laboratorio de Prácticas", ya que es precisamente en el laboratorio donde el alumno puede probar, experimentar y afianzar los conocimientos adquiridos en las sesiones teóricas.
Sin embargo, en determinadas titulaciones relacionadas con el desarrollo de las nuevas tecnologías (Ingeniería Informática, Industrial, de Telecomunicaciones), los equipos y componentes de un laboratorio suelen ser caros y su uso está restringido a que los estudiantes desarrollen el trabajo compartiendo físicamente el mismo espacio que el dispositivo con el que han de practicar: robots manipuladores, sistemas motorizados de diversa naturaleza o robots móviles, entre otros.

En la actualidad, para facilitar esta tarea se están haciendo numerosos esfuerzos por diseñar laboratorios virtuales (Maza y Ollero, 2001) que permitan a los estudiantes interactuar con dichos dispositivos reales a través de Internet o usar simulaciones de los mismos lo más realistas posibles (Jain, Howlett, Ichalkaranje y Tonfoni, 2002). La mayoría de estos laboratorios están restringidos a trabajos muy específicos, frecuentemente de control (Aranda, Dormido, Marrón, Canto y Morilla, 1993). Sin embargo, existen pocos laboratorios remotos que trabajen con dispositivos reales que tengan un cierto carácter

Publicado electrónicamente: 08/01/2010
DOI: 10.4995/RIAI.2010.01.05

generalista. Por ejemplo, sería interesante contar con un robot móvil que pudiera ser utilizado en una asignatura de Control de robots móviles para experimentar en tareas de generación de trayectorias o detección de obstáculos en entornos supervisados y, a su vez, usarse en prácticas de Programación de Sistemas en Tiempo Real, o en Diseño de Sistemas Empotrados, y todo ello utilizando un entorno que permitiese a los estudiantes trabajar tanto en el laboratorio como en casa de un modo totalmente compatible. Para alcanzar este objetivo sería necesario que el laboratorio se pudiera acceder de forma remota, pero además sería muy interesante diseñar un simulador suficientemente realista que permitiese desarrollar las prácticas incluso cuando no se dispone de conexión a Internet. En ambos casos, la práctica final debería funcionar en uno u otro modo sin tener que recompilar o realizar cambios significativos. Este formato de laboratorio no está suficientemente desarrollado aún. Su diseño e implementación presenta dificultades y su implantación supone, a día de hoy, un desafío.

Los investigadores dedicados al diseño, control y programación de robots móviles han conseguido importantes avances en el uso de simuladores (tanto *2D* como *3D*), y de arquitecturas de comunicación entre procesos (middleware), que están facilitando mucho las tareas de reutilización de código y control distribuido de sistemas formados por múltiples robots y equipos de características diversas. Ese aspecto favorece el desarrollo de aplicaciones que funcionan de la misma forma en modo real y en modo simulación.

La mayoría de los frameworks desarrollados, entre los que podemos destacar *Player/Stage/Gazebo* (Xin-qing, Wen-feng y Ding-fang, 2006) y *YARP (Yet Another Robotic Platform)* (Metta, Fitzpatrick y Natale, 2006), se basan en la interacción de procesos distribuidos mediante un modelo de comunicación cliente-servidor.

En estos entornos de desarrollo la arquitectura software se estructura en capas de diferente nivel de abstracción que interactúan entre sí. Las capas de menor nivel de abstracción (las que interaccionan con el hardware – drivers –, con el sistema operativo o con los niveles de protocolos de red), ofrecen sus servicios de forma transparente a las capas de mayor nivel de abstracción. De este modo, los diseñadores de software de alto nivel pueden dedicarse a desarrollar y mejorar los niveles de presentación, aplicación e interacción del sistema con los usuarios, mientras que los diseñadores de nuevos dispositivos electrónicos, sensores o nuevos robots pueden introducir tales elementos en la arquitectura global proporcionando simplemente nuevos drivers.

Estos frameworks son muy útiles en el ámbito de la docencia de diversas asignaturas relacionadas con la Automática y la Robótica y, de hecho, en muchos laboratorios de prácticas se usan estas herramientas, ya que proporcionan un rico material de forma gratuita. Para otras asignaturas relacionadas con la Programación Distribuida y Paralela, el diseño de Sistemas en Tiempo Real o el Diseño de Aplicaciones Orientadas a Objetos, entre otras, estas herramientas resultan excelentes ejemplos de aplicación de conceptos teóricos relacionados con lo que se podría enmarcar dentro del campo de la Ingeniería del Software.
Este artículo describe cómo se han fusionado las ideas de laboratorio remoto, middleware, mini-robots y simuladores (descritas previamente), para diseñar un laboratorio de prácticas usado en una asignatura de Programación Concurrente. Dicho laboratorio permite llevar a cabo pruebas reales aplicando los conceptos teóricos adquiridos ya que se usa la plataforma robótica BENDER 3.0. El robot se puede acceder de forma remota o simulada ya que se integra en una arquitectura distribuida.

El artículo se estructura del modo siguiente: la sección 2 resume las características de la asignatura objeto principal del estudio, en la sección 3 se describen detalladamente las arquitecturas hardware, software y de comunicación del sistema BENDER, la sección 4 muestra la aplicación docente de la plataforma y detalla el procedimiento de acceso remoto al laboratorio de prácticas. Finalmente, se aportan los apartados de evaluación de resultados obtenidos, conclusiones y desarrollos futuros.

2. CARACTERÍSTICAS DE LA ASIGNATURA PROGRAMACIÓN CONCURRENTE

Programación Concurrente es una asignatura obligatoria de segundo curso de los planes de estudios de Ingeniería Técnica en Informática de Gestión e Ingeniería Técnica en Informática de Sistemas implantados en la Universidad de Huelva, donde se estudian los conceptos básicos relacionados con la concurrencia, el paralelismo y la ejecución de procesos en entornos distribuidos que cooperan o compiten entre sí para llevar a cabo una tarea determinada (Palma, 2006).

Esta asignatura es clave para fijar las bases en la que se apoyarán otras asignaturas de cursos posteriores (tanto en segundo ciclo, como en los másteres oficiales), relacionadas con la programación avanzada de aplicaciones distribuidas, la programación de sistemas en tiempo real y sistemas críticos, y la programación de sistemas y/o agentes inteligentes, entre otras.
Estas asignaturas proporcionan al estudiante de Ingeniería Informática las competencias necesarias para diseñar y desarrollar proyectos de desarrollo, verificación e implantación de software de sistemas empotrados en el terreno de la Ingeniería del Software. Estas aplicaciones de software empotrado se pueden integrar en soluciones tecnológicas diversas, tales como, robots de servicios con capacidades cognitivas, sistemas domóticos, sistemas de vigilancia inteligentes o sistemas biométricos (Marco-Simó, 2007).

Sin embargo, pese a la importancia de la asignatura Programación Concurrente, lo más habitual es que las prácticas de esta materia consistan exclusivamente en la implementación de programas que simulan determinadas situaciones, en general alejadas de la realidad, donde se pretende ilustrar el uso de herramientas de sincronización de procesos, tanto en entornos de memoria compartida como distribuida, de una forma artificial.
Por ejemplo, es habitual ilustrar el tema sobre el uso de *semáforos* (una de las primeras herramientas de sincronización que se suele estudiar en la asignatura), usando un caso muy conocido: *"La cena de los filósofos"*, descrito ampliamente en la literatura especializada. En dicho problema cada *filósofo* se modela como un proceso que realiza dos tareas de forma cíclica: *"pensar"* y *"comer"*. Todos los *filósofos* pueden pensar al mismo tiempo, pero dado qué sólo hay *N palillos* y cada *filósofo* necesita dos *palillos* para comer, no todos pueden comer simultáneamente. Se plantea, entonces, un problema de sincronización y acceso a recursos compartidos que requieren del uso de herramientas de sincronización específicas para solucionarlo. Desde un punto de vista teórico, este ejemplo es muy descriptivo a la hora de ilustrar el problema del acceso a un

recurso compartido exclusivo de forma sincronizada (las instrucciones que acceden a dicho recurso definen una *"sección crítica"* ya que deben ejecutarse en exclusión mutua), evitando que los procesos lleguen a estados no deseados, tales como, *"de acceso simultáneo al recurso exclusivo"*, *"de postergación indefinida"* o *"de interbloqueo"*. Sin embargo, cuando los alumnos tienen que resolver un problema similar en un entorno real les resulta muy difícil abstraerse del enunciado del ejemplo teórico para aplicar la misma solución a dicho caso real.

Por tanto, para conseguir mejores resultados en la comprensión de los conceptos teóricos de la asignatura es necesario replantearse los contenidos de las prácticas en el laboratorio enfocándolos hacia la resolución de problemas en escenarios complejos, tales como, sistemas de procesamiento masivo de datos distribuidos, acceso a bases de datos distribuidas o control y tele-operación de robots, entre otros.

Por otra parte, siguiendo un temario clásico de teoría de Programación Concurrente, cualquier problema debe resolverse aplicando técnicas de concurrencia adecuadas mediante el uso de diferentes herramientas de sincronización teóricas:

- En entornos de memoria compartida suelen estudiarse *semáforos*, *regiones críticas*, *regiones críticas condicionales* y *monitores*.
- En entornos de memoria distribuida se analizan diferentes estrategias de *paso de mensajes*: mediante *buzones*, *canales*, *tipo ADA* o el estándar *MPI*, entre otros.

Otro problema consiste en la implementación real de estas herramientas de sincronización estandarizadas. La elección del lenguaje, las bibliotecas y el sistema operativo utilizado supone un aspecto importante que se debe tener en cuenta.

En memoria compartida, la implementación *Java* de un monitor es distinta a la implementación de dicho monitor en *C++* (en este caso, mediante *mutex* y *variables de condición*) o en *ADA* (mediante *objetos protegidos*).

En memoria distribuida, el intercambio de información y la sincronización de procesos pueden implementarse usando *sockets* (muy cercanos al nivel de red), *RPC (Remote Procedure Call)*, *RMI (Remote Method Invocation)* en *Java*, *paso de mensajes estilo ADA*, *OPENMPI* o *MPICH*, por ejemplo.

Debido a la complejidad y las peculiaridades de la materia, la frecuente aparición de nuevas herramientas asociadas a la Programación Concurrente y Distribuida y la constante evolución de las arquitecturas distribuidas y paralelas, el docente acaba, en muchas ocasiones, desviándose del objetivo principal a la hora de impartir la asignatura que no es otro que enseñar a los estudiantes a diseñar, implementar y verificar aplicaciones concurrentes y distribuidas que resuelvan problemas reales en entornos de trabajo multidisciplinares. Para ello es necesario desarrollar un sistema de evaluación del trabajo práctico, que valore apropiadamente su esfuerzo en el marco del nuevo sistema europeo de créditos ECTS.

Con el fin de intentar solucionar los problemas expuestos anteriormente, se ha diseñado un laboratorio de prácticas de Programación Concurrente basado en el uso de una plataforma robótica real de propósito general: **BENDER** *(**B**asic **EN**vironment for **DE**veloping **R**obotic software)*.

3. LA PLATAFORMA DE PRÁCTICAS BENDER.

BENDER es un robot móvil de propósito general diseñado y construido íntegramente en la Universidad de Huelva cuyas prestaciones se han ido incrementando a lo largo de los tres últimos cursos académicos.

Del mismo modo, los enunciados de las prácticas, el simulador de la plataforma y el modo de acceso a la misma también se han mejorado notablemente, fundamentalmente durante el último curso académico.

Atendiendo a estas mejoras se han desarrollado tres versiones estables de la plataforma:

- BENDER 1.0 (curso 2006-2007) (Pavón y Ferruz, 2007).
- BENDER 2.0 (curso 2007-2008) (Pavón y Ferruz, 2009).
- BENDER 3.0 (curso 2008-2009).

BENDER forma parte de un entorno de experimentación generalista ya que posee una arquitectura software abierta que permite integrar nuevos elementos en función de las necesidades de los usuarios y de los desarrolladores. Puede ser utilizado como plataforma de experimentación de algoritmos de control de vehículos autónomos, de tele-operación o de mejora de la comunicación entre procesos, entre otros.

El laboratorio de prácticas en el que se integra BENDER puede ser utilizado en asignaturas tan diversas como Robótica, Sistemas en Tiempo Real o Programación Concurrente (el caso que nos ocupa), siempre y cuando los enunciados de los trabajos prácticos se adecúen a la interfaz de la arquitectura software distribuida que permite el funcionamiento del vehículo.

En la actualidad, la plataforma robótica se utiliza como base fundamental de los laboratorios de prácticas de las siguientes asignaturas: Programación Concurrente y Diseño de mini-vehículos autónomos inteligentes, y en proyectos fin de carrera tanto de primer como de segundo ciclo de la titulación de Ingeniería Informática.

3.1 Arquitectura hardware de BENDER.

La Figura 1 muestra la versión 3.0 de la plataforma BENDER.

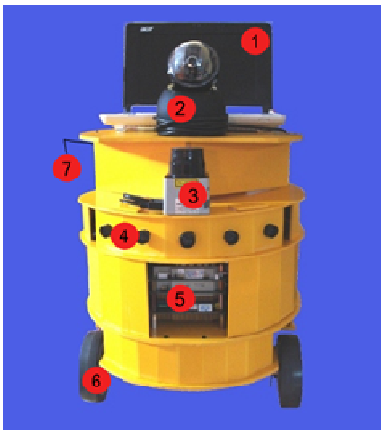

Figura 1. Versión actual de la plataforma BENDER.

En la Figura 1 se señalan los principales componentes del robot móvil desde el punto de vista mecánico-electrónico:

1) **PC Ultraportátil (Notebook) ACER ASPIRE ONE para tareas de visión artificial**. Este PC se denomina "bender-eyes" dentro de la arquitectura distribuida de ordenadores del sistema.

   Intel Atom N270 a 1.6 GHz. – HDD de 160 Gb. – RAM de 1 Gb. – 8.9" (22.6 cm) 1024x600 LED TFT – Ethernet 10/100 Mbit – 802.11b/g WLAN – 3 puertos USB 2.0.

2) **WebCam Logitech Sphere MP con sistema PTZ** (Pan and Tilt and Zoom).

3) **HOKUYO Scanning Laser:** 240° de área de barrido - Alta resolución de 0.36°.

4) **Anillo de sónares**: 5 sónares delanteros y 2 sónares traseros de tipo *SFR02* con comunicación vía *I2C* bus o serie, frecuencia de 40kHz y rango de medidas en el intervalo [15cm, 6 m].

5) **Embedded PC-104.** Este PC se denomina "fry".

   Intel LV Pentium® III 800, RAM 512 Kb., Memory Card 1Gb., con ampliación de puertos *USB* y *Firewire*.

6) **Sistema de tracción diferencial***:* con controlador *Devantech MD23* en puente H de dos motores de corriente continua con encoders *EMG30*.

7) **Punto de acceso inalámbrico** *802.11b/g* con puerto Ethernet para conectar el *PC-104* "fry" a la red.

Además de los dispositivos visibles en la Figura 1, el robot dispone de una brújula digital con comunicación *I2C* y un sistema de alimentación basado en una batería de plomo de *12V* y *7AH*. El PC-Ultraportátil "bender-eyes" se alimenta con una batería propia, lo que permite que el sistema de visión pueda ser utilizado de forma independiente al resto de componentes.

Todos los dispositivos que utilizan el protocolo *I2C* (Paret, 1997) para comunicarse (controlador *MD23* de tracción, anillo de sónares y brújula digital) se conectan a "fry" usando los conectores USB, mediante un módulo conversor *USB-I2C* de tal modo que los dispositivos del sistema pueden controlarse, fácilmente, a través de puertos serie virtuales.

A continuación, se enumeran las conexiones de puertos series virtuales asociados a cada dispositivo o grupo de dispositivos.

- Puerto **/dev/ttyACM0** → Puerto serie de comunicación para el dispositivo *Hokuyo Scanning Laser*.
- Puerto **/dev/ttyUSB0** → Puerto serie de comunicación para la brújula digital. La brújula digital envía y recibe datos a través de un dispositivo maestro *USB-I2C* con el que comparte el bus *I2C*.
- Puerto **/dev/ttyUSB1** → Puerto serie de comunicación para el sistema de tracción *MD23*, que proporciona información sobre la lectura de los encoders y permite enviar comandos de actuación a los motores de forma independiente. Como en el caso anterior, la comunicación se establece utilizando un adaptador *USB-I2C*.
- Puerto **/dev/ttyUSB2** → Puerto serie para comunicación del grupo de sónares que forman el anillo. En este caso, los siete sónares comparten el bus *I2C* y el conversor *USB-I2C* actúa como dispositivo maestro. Cada sónar *SFR02* del anillo se identifica mediante una dirección distinta dentro del bus. Es posible enviar un comando de sólo emisión de ultrasonido, de sólo recepción o de estimación de la medida completa.

La interfaz *USB-I2C* se implementa mediante un chip *FTDI FT232BM USB* cuyos drivers están disponibles para Windows, Linux y Mac-OS. Todos los dispositivos que se conectan a "fry" mediante dicha interfaz utilizan un protocolo de comunicación basado en el envío y recepción de tramas de bytes a través del puerto serie.

La
Figura 2 muestra de forma resumida cómo se interconectan todos estos dispositivos electrónicos al PC empotrado "fry".

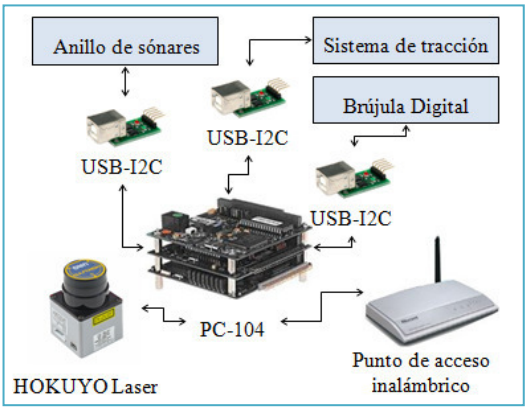

Figura 2. Conexión de los componentes al PC-104 "fry".

3.2 Arquitectura software de BENDER.

El software de control de BENDER se estructura mediante una arquitectura distribuida formada por módulos independientes (diseñados usando el concepto de *capa de servicio*), que pueden ejecutarse en diferentes máquinas. Los módulos que interactúan con los dispositivos físicos del robot deben necesariamente ser ejecutados en el PC "fry" mientras que el resto de procesos pueden ejecutarse en otras máquinas remotas.

Para llevar a cabo las tareas de visión artificial, parte de los procesos que tienen que ver con la captura de imágenes y manejo del *PTZ* de la cámara deben ejecutarse en el PC "bender-eyes".

La comunicación entre procesos se lleva a cabo utilizando el framework middleware YARP; de este modo, el diseñador de la arquitectura puede abstraerse de los niveles y protocolos de red, entendiendo el sistema distribuido como una red de puertos YARP que comparten la información usando un modelo cliente-servidor.

YARP determina por defecto que la comunicación entre los procesos se hará mediante *TCP* independientemente de la ubicación de los mismos. El programador puede seleccionar otros protocolos de comunicación de modo que dos procesos que se ejecutan en la misma máquina pueden intercambiar información mediante el uso de un área de memoria compartida mejorando la eficiencia de la aplicación, mientras que dos procesos que se ejecutan en máquinas distintas deberán conectarse usando algún protocolo de red: *TCP*, *UDP* o *multicast*. Una vez hecha la conexión la gestión de la comunicación es transparente al usuario de la red YARP. Cuando los procesos necesitan compartir información utilizan puertos con un nombre único dentro de la red y adoptan el comportamiento de *servidor* o *cliente*.

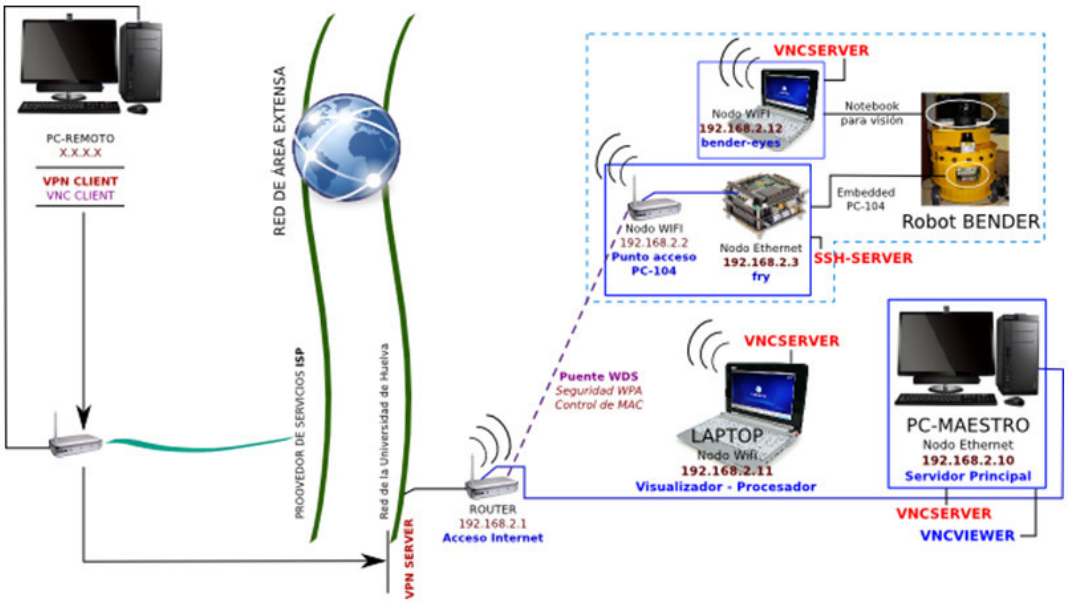

Figura 3. Arquitectura de comunicaciones del laboratorio remoto de prácticas

Los *servidores* abren puertos que quedan registrados en la red YARP a través del *servidor* YARP. Los clientes abren sus propios puertos que también quedan registrados por el *servidor* YARP y realizan las conexiones a los puertos de los servidores con los que desean intercambiar información. La conexión efectiva es llevada a cabo por el *servidor* YARP que se ejecutará en una de los nodos de la red.

El uso de la red YARP permite que BENDER pueda ser instalado en cualquier sala de ordenadores que tenga instalado el sistema operativo Linux y la última versión de YARP. Normalmente, los ordenadores de la sala suelen formar una red de área local, por lo que el servidor YARP sólo tiene que ser ejecutado en uno de los nodos. BENDER puede conectarse a esta red de área local mediante un punto de acceso inalámbrico, de este modo los alumnos pueden interactuar con el sistema desde su puesto.

Los computadores que forman parte de la arquitectura BENDER 3.0 tienen las siguientes funciones, ver Figura 3:

- **PC-104 "fry"** → Permite la ejecución de los procesos de control de bajo nivel que interactúan con los sensores y actuadores del robot móvil.
- **PC ultra-portátil "bender-eyes"** → Permite la ejecución de los procesos que llevan a cabo el control de bajo nivel de la cámara y el *PTZ*. Realiza también tareas básicas de procesamiento de imágenes.
- **PC de sobremesa Intel Core 2 Duo "PC-MAESTRO"** → Realiza las veces de servidor de conexiones remotas y servidor web para gestión de sesiones de prácticas remotas.

  Además, puede ejecutar procesos que lleven a cabo el análisis o manejo de datos obtenidos de los sensores para establecer pautas de comportamiento y selección de acciones en función de los objetivos a alcanzar por el robot móvil. Por ejemplo, si se desea diseñar un experimento de evitación de obstáculos, los procesos que analizan la información proveniente del láser o del anillo de sónares, e implementan el algoritmo de evitación propiamente dicho, podrían ejecutarse en "PC-MAESTRO". En este nodo también se ejecutarían aquellos algoritmos que requieran la gestión de grandes archivos de *log* o deban realizar tareas relacionadas con el tratamiento de secuencias de imágenes o aprendizaje de datos.

  PC portátil "LAPTOP" → Realiza tareas auxiliares: procesos de alta carga computacional y simulación gráfica de los experimentos.

La Figura 4 muestra un esquema gráfico de la arquitectura software distribuida.

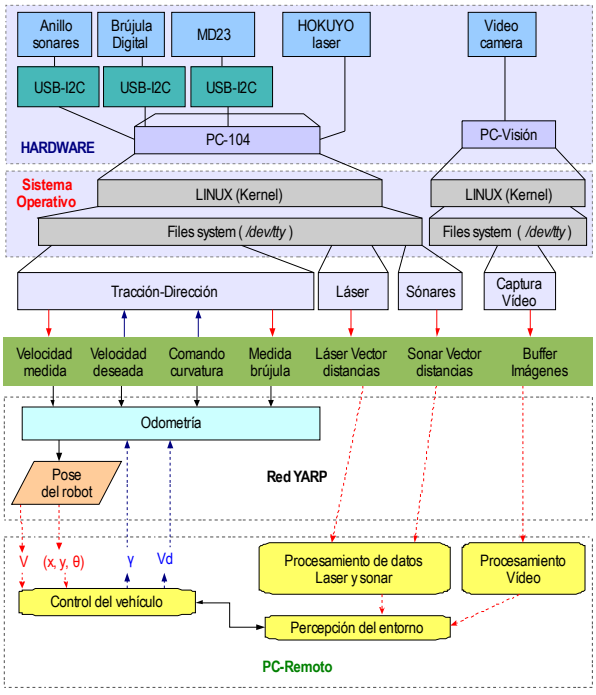

Figura 4. Arquitectura software en detalle.

Esta arquitectura admite la posibilidad de sustituir el conjunto de procesos que interactúan con el robot real por un simulador del comportamiento del vehículo, siempre que dicho simulador proporcione los datos a la red YARP usando los mismos nombres de puerto que los procesos que controlan el robot real.
Este rasgo es muy interesante, ya que si la red de comunicaciones falla o los alumnos o investigadores que trabajan con la plataforma desean trabajar en casa o en la oficina, de forma local, pueden hacerlo sin tener que reescribir o recompilar los programas que ya forman parte de la aplicación concreta que trabaja sobre la arquitectura software de BENDER.

3.3 Arquitectura de comunicaciones para llevar a cabo un servicio de laboratorio de prácticas remoto.

Como ya se ha comentado se dispone de un simulador para facilitar el uso de BENDER de forma local, ver Figura 5.

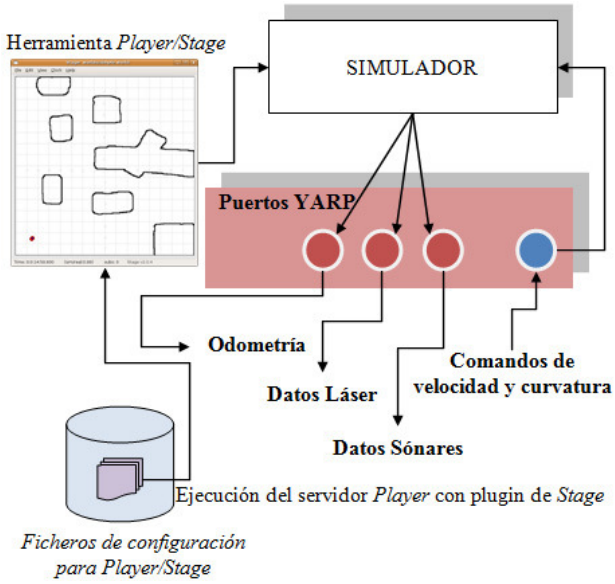

Figura 5. Funcionamiento del simulador.

Dicho simulador está basado en la herramienta de simulación de robots *Player/Stage* y proporciona los datos básicos de odometría, lectura de sónares y lectura de los datos del láser mediante la interacción del robot simulado con un mundo en dos dimensiones, también simulado. El uso de este simulador es esencial para que los alumnos puedan desarrollar sus prácticas en casa sin necesidad de una conexión a Internet.

Por otra parte, para aquellos investigadores o alumnos de proyecto que prueban nuevos algoritmos de control, planificación o aprendizaje sobre la plataforma robótica, el simulador resulta de gran utilidad para realizar el diseño rápido de prototipos que permitan un proceso de verificación del código más fácil. Sin embargo, para comprobar el funcionamiento de las aplicaciones finales, el uso del robot móvil real es imprescindible. En estos casos, el estudiante o investigador debe estar físicamente en el mismo espacio en el que se encuentra el vehículo, a no ser que se proporcionen los medios apropiados para interactuar con él de forma remota. Por ello, durante el curso académico 2008-2009 se ha implantado un sistema de acceso remoto al robot móvil real utilizando la arquitectura de comunicaciones de la Figura 3.

En este caso, el robot se ubica físicamente en un despacho situado en el edificio del Departamento de Tecnologías de la Información de la Universidad de Huelva. Esto implica que se dispone de un acceso a Internet desde la red corporativa de la universidad a través de una dirección IP fija asignada en el ámbito de una subred dentro de la red corporativa. Como se observa en la Figura 6, cuando se desea conectar de forma cableada más de un ordenador a la red corporativa es necesario utilizar un dispositivo tipo *switch* o *router*. El uso de un *router* implica la creación de una nueva subred dentro de la subred departamental.

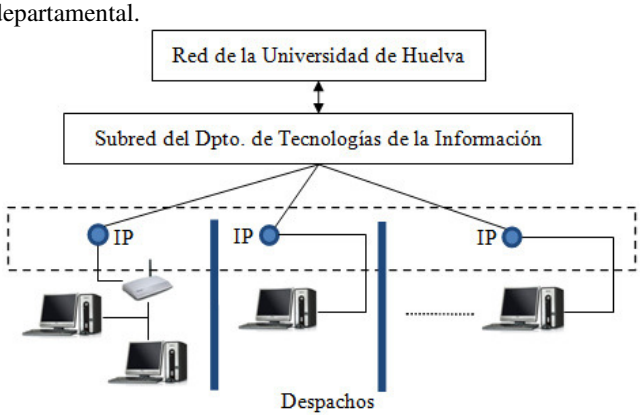

Figura 6. Esquema de conexiones para los despachos de un departamento de la Universidad de Huelva.

Además el sistema BENDER necesita utilizar conexiones inalámbricas, por lo que se ha elegido un *router* inalámbrico que permite la conexión de 4 equipos de forma cableada y el resto usando una red *wifi* propia del sistema. La subred del sistema BENDER proporciona direcciones *IP* en el rango *192.168.2.1 – 192.168.2.255* para la conexión de los diversos equipos, tanto de forma cableada como inalámbrica. Como muestra la Figura 3, se ha mapeado la dirección real de la subred departamental sobre la dirección local PC-MAESTRO, por tanto, desde fuera de la red corporativa sólo es posible acceder a dicho ordenador, utilizando una conexión segura *ssh* o un cliente de escritorio virtual remoto (*vncviewer*). El ordenador remoto debe disponer de una conexión *VPN* que autorice su integración temporal en la red corporativa de la Universidad de Huelva mediante un certificado digital expedido por el órgano administrativo competente, que asegura que se cumplen los parámetros generales de seguridad.
En cualquier caso, desde los laboratorios de prácticas ubicados en la Escuela Politécnica Superior de la Universidad de Huelva, sí es posible acceder al laboratorio remoto, por encontrarse dichos laboratorios dentro de la red corporativa. Esto facilita la realización de prácticas de forma semi-presencial. El modo básico de trabajo es el siguiente:

- A través del PC-MAESTRO se puede acceder mediante escritorio remoto al resto de ordenadores del sistema. La función de cada uno de los PC's, denominados *nodos de procesamiento* se ha explicado en la sección 3.2.
- El nodo PC-MAESTRO se conecta a*l router principal* de forma cableada.
- Los nodos de procesamiento LAPTOP y *bender-eyes* se comunican con el *router principal* utilizando la conexión inalámbrica. Sólo se permite que dichos PC's se conecten vía *wifi* al *router principal*, mediante identificación a través de *MAC* (*Media Access Control*).
- El acceso a "fry" se hace mediante una conexión segura *ssh*. El PC "fry" se conecta a la red a través de un punto de acceso inalámbrico. Dicho punto de acceso se conecta al *router principal* usando el protocolo *WDS (Wireless*

Distribution System), aplicando seguridad *WPA (WiFi Protected Access)*.

Desde un punto de vista práctico, el usuario de la plataforma remota BENDER sólo necesita conocer la dirección *IP* asignada al *router principal* por la red corporativa de la Universidad de Huelva. La configuración *NAT (Network Address Translation)*, del *router principal* permitirá mapear los puertos que se deseen al nodo de procesamiento PC-MAESTRO. En el PC-MAESTRO se deben crear tantos usuarios como sean necesarios para permitir una interacción multiusuario remota con la plataforma BENDER.

Desde un punto de vista software, la plataforma BENDER debe proporcionar un conjunto de procesos que controlen el acceso al robot móvil por parte de varios usuarios. Las operaciones de lectura sobre puertos son compatibles entre sí, sin embargo, las operaciones de escritura o de prueba de un determinado experimento deben realizarse en exclusión mutua. Este funcionamiento muestra a los alumnos de la asignatura Programación Concurrente las dificultades y desafíos del desarrollo de software que ha de funcionar de manera distribuida y en tiempo real, por lo que el estudio de la arquitectura de comunicación es, en sí mismo, una valiosa fuente de aprendizaje y entrenamiento.

Para solventar el problema del acceso compartido actualmente se está trabajando en un sistema de gestión de sesiones a través de la web. Ya se ha desarrollado un primer proyecto fin de carrera que resuelve la parte del problema relativa al control de acceso de los usuarios a la arquitectura distribuida.
La siguiente sección muestra el uso académico de la plataforma, fundamentalmente en Programación Concurrente y presenta las extensiones propuestas al proyecto mencionado.

4. USO ACADÉMICO DE BENDER.

BENDER se utiliza en el laboratorio de prácticas de la asignatura Programación Concurrente con un objetivo fundamental: mostrar a los estudiantes un sistema distribuido real cuyo control y funcionamiento se basa en programar adecuadamente las herramientas de sincronización y comunicación de modo que el sistema realice las tareas asignadas cumpliendo las dos propiedades básicas de la programación concurrente: seguridad y vivacidad.
Repasando los cursos académicos en los que BENDER se ha utilizado como herramienta esencial se puede observar que la plataforma ha ido evolucionando y mejorando para ofrecer un mejor servicio y funcionalidad.

- Durante el curso académico 2006-2007, mientras se diseñaba y construía la primera versión de la plataforma, los alumnos comenzaron a trabajar con ella a través de la realización de prácticas que consistían en simular el funcionamiento interno de los dispositivos *I2C* del robot. Cada dispositivo se simulaba mediante hilos *POSIX* en *C* donde el recurso a compartir era el bus *I2C*.
- Con la versión 2.0 en funcionamiento, los estudiantes del curso académico 2007-2008 pudieron implementar sus prácticas usando el robot real como plataforma de pruebas. En este caso, el objeto de las prácticas era diseñar una extensión que mejorase la funcionalidad de la arquitectura distribuida software añadiendo un módulo de análisis y visualización de los datos obtenidos con el anillo de sónares. En la sección 4.1 se especifica detalladamente el enunciado práctico propuesto.
- Durante el curso académico 2008-2009, los estudiantes han realizado una práctica consistente en adquirir los datos provenientes del sistema de tracción del robot y llevar a cabo los cálculos de odometría necesarios para mostrar la trayectoria descrita por el vehículo en tiempo real.

Por otra parte, la versión 3.0 también se ha utilizado como herramienta práctica en el curso de libre configuración "Diseño de mini-vehículos autónomos inteligentes".

Además, se ha completado la primera fase del sistema de ejecución remota de experimentos a través de la web en el ámbito del proyecto fin de carrera mencionado en la sección 3.3.

También, en el marco de desarrollo de otro proyecto fin de carrera, se está trabajando en el diseño integrado de una interfaz de usuario gráfica que facilite el control de BENDER, la gestión de puertos YARP, el acceso al simulador, la representación gráfica de resultados y el control del sistema de visión de forma transparente y sencilla.

4.1 Ejemplo de ejecución del plan docente de prácticas de Programación Concurrente durante el curso 2007-2008.

Esta sección muestra, en detalle, cómo se realiza una práctica usando la plataforma descrita en los apartados anteriores. El objetivo fundamental de la práctica, diseñada durante el curso académico 2007-2008, consistía en extender la funcionalidad de BENDER, proporcionando un conjunto de programas que analizaran y visualizaran los datos provenientes del anillo de sónares. Se pretendía que los estudiantes se entrenaran aplicando las herramientas de sincronización estudiadas en teoría (en concreto, una solución basada en el modelo *Productor-Consumidor*), en un marco complejo de trabajo real.

Documentación necesaria para el desarrollo de la práctica propuesta.
Junto con el enunciado, los alumnos recibieron una documentación detallada sobre el funcionamiento de la red YARP y el nombre, características y uso de sus puertos. Para llevar a cabo la lectura o escritura de información sobre un puerto YARP, además de conocer el nombre del mismo, es necesario conocer el formato y el orden de la secuencia de datos que se transmitirá. Para ello, YARP pone a disposición de los desarrolladores un conjunto de bibliotecas para programar interfaces personalizadas en *C++* que permiten llevar a cabo la serialización de dichos datos. Se proporcionó a los estudiantes el código fuente de las clases que serializaban los datos y dos aplicaciones: un simulador para facilitar el trabajo en casa y un visualizador gráfico de los datos del anillo de sónares. El simulador leía un fichero de texto que guardaba un conjunto de datos reales obtenidos del vehículo con el siguiente formato:

Orientación	*S1*	*S2*	*S3*	*S4*	*S5*	*S6*	*S7*

Donde *"Orientación"* representaba la orientación del vehículo respecto de una orientación inicial de 0 grados y *Si* representaba la lectura del sónar *i* del anillo de sónares. Cada tupla era enviada, a intervalos fijos, a un conjunto de puertos YARP, nombrados como en el vehículo real, simulando así el comportamiento del robot.

Objetivo de la práctica.

La aplicación del alumno debía recibir dichos datos, almacenarlos, procesarlos desde el punto de vista estadístico y

reenviarlos a la aplicación de visualización, siguiendo el esquema clásico del *"Productor-Consumidor"*, con un proceso productor y dos consumidores (uno de ellos denominado *Estadístico*). Dicho esquema es ampliamente utilizado en la resolución de problemas de Programación Concurrente basados en la utilización de un *buffer* limitado compartido con acceso en exclusión mutua. La Figura 7 muestra el esquema básico de funcionamiento de la aplicación a implementar por los estudiantes de la asignatura.

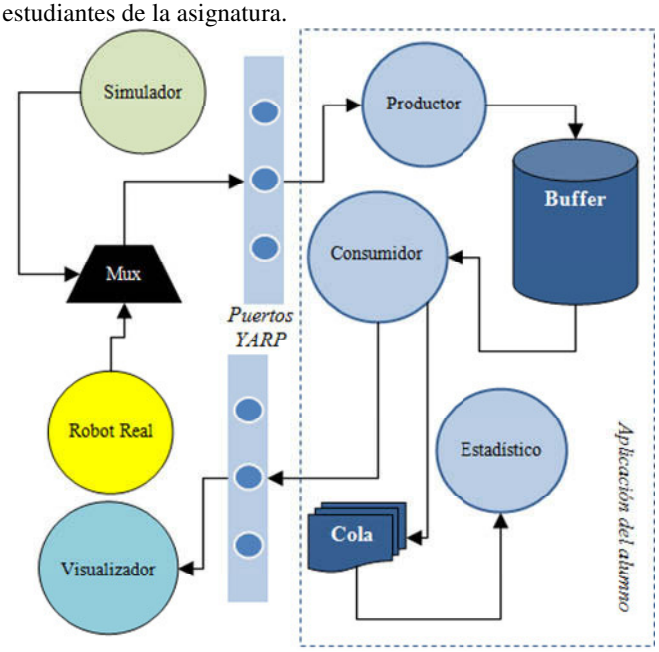

Figura 7. Esquema de la aplicación práctica a implementar por los alumnos de Programación Concurrente.

El objeto *Mux* de la Figura 7 representa la posibilidad de probar el programa resultado, tanto en el simulador como en el robot real, sin necesidad de cambiar ni recompilar ningún módulo.

Ejecución de las pruebas de la práctica.
La Figura 8 muestra el resultado de la ejecución de la práctica resuelta. Como puede observarse, los estudiantes utilizan una aplicación que recibe los datos del vehículo a través de la red YARP y otra aplicación, en este caso un programa gráfico basado en GTK, que utilizando la misma red, recibe los datos procesados de la aplicación del alumno para representar el alcance de los 7 dispositivos que componen el anillo de sónares del vehículo.

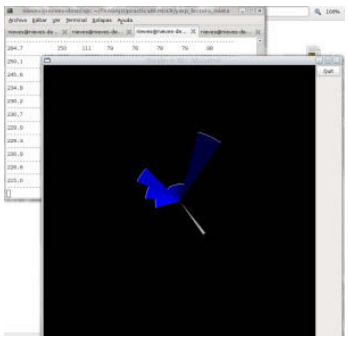

Figura 8. Resultado de la ejecución de la práctica.

Las pruebas sobre el robot real pueden desarrollarse utilizando el sistema de red de BENDER, descrito en la sección 3.3, a través de conexiones remotas VNC. Si los comandos son de lectura de puertos todos los estudiantes pueden tener acceso simultáneo al vehículo. Si los comandos son de escritura de puertos el sistema YARP emite un mensaje que especifica la imposibilidad de realizar la conexión cuando el canal de comunicación está ocupado. Esto es necesario ya que un motor no puede ser movido usando diferentes comandos de control a la vez.

4.2 Servidor de sesiones de prácticas remotas a través de la web.

Algunos de las dificultades derivadas de la ejecución remota de programas sobre la plataforma BENDER generalmente tienen que ver con las directivas de seguridad de red impuestas por la universidad y con el control de aquellas acciones que puedan ocasionar daño físico a la propia plataforma. Para resolver estos problemas, se ha propuesto el diseño de un *servidor de sesiones de prácticas remotas* mediante acceso web. El diseño de dicho sistema se ha planteado en dos fases:

- Fase de diseño de acceso a web para identificación de estudiantes, carga de códigos fuentes y compilación remota de las prácticas, ya concluida.
- Fase de ejecución remota de las prácticas y volcado de resultados a través de Internet, actualmente en desarrollo.

Durante el segundo cuatrimestre del curso 2009-2010 se espera poner en funcionamiento el sistema completo de modo que se puedan diseñar módulos que se incorporen al conjunto de aplicaciones software distribuidas de BENDER tras una fase de análisis automático que garantice la seguridad de dicho módulo. La Figura 9 muestra la arquitectura básica de funcionamiento del servidor de sesiones remotas de prácticas descrito.

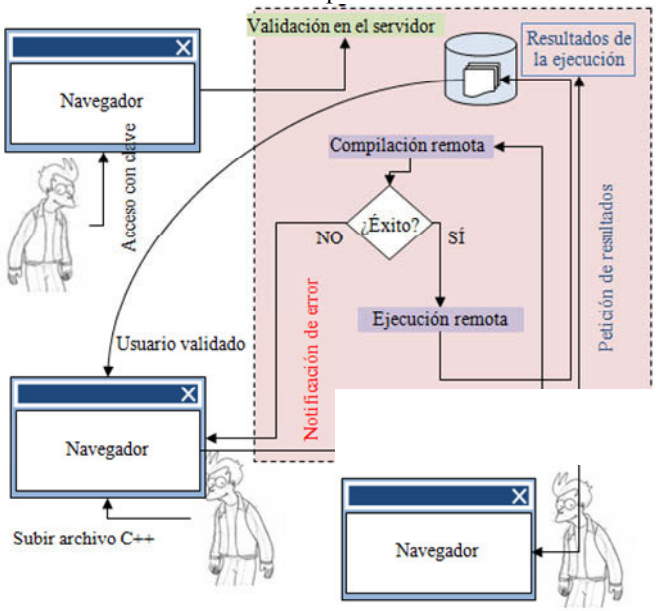

Figura 9. Esquema de funcionamiento del servidor remoto.

Los pasos a seguir para compilar y ejecutar una práctica se enumeran a continuación. En la aplicación diseñada, los puntos 1, 2, 3 y 4 corresponden a la primera fase del diseño del servidor de sesiones, mientras que en el resto de los puntos se está trabajando en la actualidad, encontrándose el sistema global en una fase muy avanzada.

1) Validación de acceso al sistema a través de un navegador web. El alumno se identifica mediante un nombre de usuario y una contraseña. La primera vez que el alumno accede al sistema se genera una contraseña

automáticamente que debe ser cambiada por el estudiante antes de realizar cualquier otra tarea. Cuando un alumno se da de alta en el sistema se crea automáticamente una carpeta en el servidor usando como identificador el DNI. El profesor tiene acceso a la ficha del alumno y puede hacer un seguimiento estricto de sus avances.

2) El sistema muestra una ficha con los datos del alumno y presenta un formulario que permite "subir" documentos de tipo texto, C++ y PDF.
3) Para poder ejecutar prácticas de forma remota es necesario subir el archivo C++ con el código de la misma. En este paso el docente puede decidir cuántos ficheros fuentes son necesarios, si se subirá una aplicación completa o sólo una parte que se integre en un módulo previamente compilado.
4) En el "PC-REMOTO" del *sistema de red* BENDER, ver Figura 3, se inicia una sesión de compilación en la carpeta de prácticas del alumno que solicita dicha acción. Previamente, se analiza automáticamente el contenido del archivo fuente para evitar la ejecución de código malintencionado o llamadas al sistema de carácter peligroso. Si la compilación tiene éxito el programa pasa a una cola de prácticas pendientes de ejecución.
5) Un proceso del "PC-REMOTO" denominado *"ejecutor"* analizará continuamente el estado de la cola de prácticas pendientes. Tomará el programa más antiguo y lo ejecutará como un módulo más de la arquitectura distribuida del sistema. Para evitar choques y daños físicos sobre la plataforma, ciertos procesos de alta prioridad examinarán las acciones propuestas por el proceso *"práctica"* antes de que se lleven a cabo.
6) El resultado del experimento quedará grabado en archivos de texto que mostrarán todas las variables a examinar. Si es necesario se activará un sistema de captura de imágenes que permita grabar la prueba y almacenar el vídeo en un formato adecuado para que el alumno pueda descargárselo posteriormente. El alumno podrá ser informado de la ejecución de su práctica tanto a través de la web como por correo electrónico.
7) Finalmente, la práctica pendiente será borrada de la cola y los archivos de resultados se almacenarán en la carpeta del alumno.

Cabe resaltar que este sistema de entrega y ejecución de prácticas podría adaptarse a cualquier asignatura definiendo simplemente el objetivo de la práctica y añadiendo a la arquitectura software los procesos de control auxiliares necesarios. Estos procesos deberían ser implementados por los docentes de la asignatura.
De este modo se consigue un laboratorio virtual remoto de carácter bastante generalista.

5. RESULTADOS ACADÉMICOS DE LOS ALUMNOS

En esta sección se comparan los resultados académicos obtenidos por los estudiantes a lo largo de los últimos cuatro cursos académicos. Sólo durante el curso 2005-2006 no se ha usado la plataforma BENDER para la realización de las prácticas, en este caso, los ejercicios se basaron en la simulación de problemas clásicos teóricos.

La Figura 10 muestra la evolución de los resultados académicos de los alumnos a lo largo de los cuatro últimos cursos. (Se han utilizado los datos de la convocatoria de Junio).

En la Tabla 1 pueden observarse estos resultados de forma numérica. Los valores de las columnas *NP* (no presentado), *Suspenso* y *Aprobado* se representan mediante porcentajes. La columna *Nº de alumnos* se refiere al total de alumnos matriculados en cada curso tanto en la titulación de *Ingeniería Técnica en Informática de Gestión* como de *Sistemas*. La Tabla 2 muestra los resultados desglosados, tanto por grupos y titulación como por calificación obtenida. En este caso, T hace referencia a la titulación (G = *Gestión* y S = *Sistemas*), Gr se refiere al grupo (*T1* o *T2*). Las calificaciones se agrupan en NP = *No Presentado*, Su = *Suspenso*, A = *Aprobado*, N = *Notable*, S = *Sobresaliente* y MH = *Matrícula de Honor*.

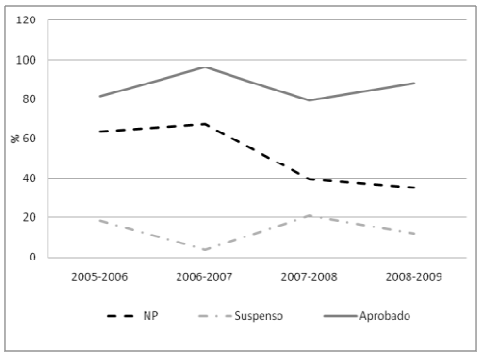

Figura 10. Evolución de los resultados académicos.

Tabla 1. Resultados académicos globales.

Curso	Nº de alumnos	NP	Suspenso	Aprobado
2005-2006	189	63.62	18.46	81.54
2006-2007	165	67.24	3.73	96.27
2007-2008	138	39.64	20.76	79.24
2008-2009	77	35.20	11.94	88.06

Tabla 2. Resultados académicos desglosados.

Curso	T	Gr	NP	Su	A	N	S	MH
2005/2006	G	T1	54.54	18.18	14.54	10.90	1.82	0
		T2	63.63	20	10.90	5.45	0	0
	S	T1	63.15	21.05	10.52	2.63	2.63	0
		T2	73.17	14.63	7.32	2.44	2.44	0
2006/2007	G	T1	79.62	3.70	7.41	7.41	1.85	0
		T2	74.35	0	10.25	10.25	2.56	2.56
	S	T1	50	6.25	25	15.62	0	3.12
		T2	65	5	2.5	20	2.5	5
2007/2008	G	T1	43.90	29.26	9.76	12.19	2.44	2.44
		T2	40.47	16.66	23.80	14.28	4.76	0
	S	T1	34.54	16.36	5.45	30.90	10.90	1.82
2008/2009	G	T1	43.33	10	10	26.66	3.33	6.67
		T2	38.46	11.53	15.38	23.07	3.85	7.69
	S	T1	23.80	14.28	0	52.38	9.523	0

En la Figura 10 cabe destacar la evolución positiva del número de alumnos presentados desde que se comienza a usar la plataforma como herramienta práctica. Hay que tener en cuenta que BENDER es el eje central no sólo del laboratorio sino de los ejercicios prácticos que se llevan a cabo en las clases de teoría programadas en la guía ECTS de la asignatura. Esto conlleva que los estudiantes se familiarizan con la plataforma desde el inicio del curso y adquieren las competencias necesarias para asociar el contenido teórico de la asignatura con los problemas reales que se pueden resolver utilizando las técnicas aprendidas.

Por otra parte, a medida que aumenta el número de alumnos presentados lógicamente aumenta el número de alumnos que

suspenden la materia en primera convocatoria. Sin embargo, este aumento es menor en comparación con el aumento del número de aprobados en general y de calificaciones altas en particular, tal y como se muestra en la Tabla 2.

6. CONCLUSIONES

Al analizar los resultados obtenidos tras usar BENDER en la asignatura objeto de estudio, se concluye que:

- Con el uso de la plataforma robótica los resultados académicos de los estudiantes han mejorado respecto a cursos anteriores y ellos mismos consideran que trabajar en un entorno real supone un desafío que les da la oportunidad de profundizar en los contenidos de la asignatura. Esto implica una mayor dedicación de tiempo y esfuerzo a la misma.
- Antes de que estuviera en funcionamiento el sistema de red de la plataforma, las pruebas en el robot real resultaban bastante complicadas, sin embargo, el acceso remoto a BENDER vía VNC ha facilitado el proceso de prueba de las aplicaciones finales.
- El hecho de disponer de un único robot a compartir por muchos estudiantes supone un mayor esfuerzo de planificación para la ejecución concurrente de prácticas. Para abordar esta cuestión se encuentra en una fase muy avanzada de desarrollo el *servidor remoto de prácticas* que proporcionará una herramienta automática de gestión de experimentos sobre la plataforma BENDER.
- El uso de una arquitectura distribuida basada en YARP permite que los diseñadores, tanto de la propia arquitectura como de los experimentos prácticos, se abstraigan de las cuestiones relacionados con el nivel de red y se centren en construir programas que manejen los canales de comunicación de un modo eficiente.
- El uso de YARP es especialmente útil cuando se desea construir una aplicación que trabaje tanto con el robot real como con el simulador del robot de forma transparente. Mientras se respete la nomenclatura de los puertos las aplicaciones diseñadas serán compatibles en ambos casos, sin necesidad de hacer cambios sobre el código fuente ni recompilar. Esta característica es imprescindible para que los estudiantes puedan trabajar en casa incluso si no disponen de acceso a Internet.
- La adición de nuevos dispositivos y robots al sistema es sencilla debido a que tanto la arquitectura software como de comunicaciones permiten un grado muy elevado de heterogeneidad de los equipos que conforman la red.
- Desde el punto de vista económico, BENDER es un robot móvil con muchas prestaciones, de propósito general y de bajo coste. Aproximadamente el coste total hardware de la plataforma es reducido (unos 4000 euros), si se compara con otras plataformas comerciales tales como robots *irobot* o *Pionner*, muy utilizados para la experimentación en entornos *indoor*.

7. DESARROLLOS FUTUROS

Actualmente se está mejorando la parte mecánica del sistema y se ha incrementado el número de computadores transportados por el vehículo que forman parte de la arquitectura distribuida. Esto aumenta la capacidad de procesamiento del robot cuando ha de realizar tareas de forma totalmente autónoma. Además, se ha incluido un sistema de visión estéreo que permite acceder a los pares de imágenes usando los componentes apropiados del sistema YARP y OpenCV. En cuanto al trabajo futuro a desarrollar se pueden especificar los siguientes objetivos:

- Puesta a punto del *servidor remoto de prácticas* antes del segundo cuatrimestre del curso académico 2009-2010.
- Llevar a cabo un proceso de divulgación del sistema a nivel departamental para que sea conocido y utilizado en diversas asignaturas.
- Ampliar la plataforma añadiendo nuevos robots móviles.
- Proponer aplicaciones nuevas en el ámbito de la *Robótica Cognitiva*.

AGRADECIMIENTOS

Este trabajo ha sido realizado, parcialmente, gracias al apoyo del *Vicerrectorado de Tecnologías, Innovación y Calidad* de la *Universidad de Huelva*.

REFERENCIAS

Aracil, R., Balaguer C., Armada, M. (2008). Robots de servicio. *RIAI* **Vol.5**, Num. 2. pp 6-13.

Aranda, J., Dormido, S., Marrón, J. L., Canto, F., (1993). Study and Building of a Computer. (P. Nobar y W. Kainz, Eds.) *Computer Based Learning in Science* , 561-566.

Casals, A. (2007). Robots de servicios y personales: Aspectos tecnológicos que dificultan su progreso. *Asociación Española de Robótica y Automatización de Tecnologías de la Producción (AER-ATP)*.

Gates, B. (2007). A robot in every home. *Scientific American*, pp 58-65.

Jain, L.C., Howlett, R.J., Ichalkaranje, N.S., Tonfoni, G. (2002). Virtual Environments for Teaching & Learning. *World Scientific-Series on Innovative Intelligence*. **Vol.1**, 2002.

Marco-Simó, J.M. (2007) ¿Podemos darle la vuelta a la enseñanza del desarrollo de software? *Jornadas de ENseñanza Universitaria de la Informática (JENUI)*, 15-17 Julio, Teruel.

Maza I., Ollero, A., Hemero (2001): A Matlab-Simulink toolbox for robotics. *First Workshop on Robotics Education and Training (RET)*. Weingarten, Germany, Imatge i Publicacions UPC, pp 43-50.

M.E.C. (2003). *Ministerio de Educación y Ciencia. Real Decreto 1125/2003*, B.O.E. num. 224, pp 34355-34356.

Metta, G., Fitzpatrick, P., Natale, L. (2006). YARP: Yet Another Robotic Platform. *International Journal of Advanced Robotics Systems (ARS)*. pp 43-48, vol. 3, number 1.

Pagani, R. (2002). *El Crédito Europeo y el Sistema Educativo Español. Informe Técnico*.

Palma, J. T. (2006). *Programación Concurrente*. Ed. Paraninfo, Spain, 2006.

Paret, D. (1997). *The I2C Bus from Theory to Practice*. John Wiley and Son.

Pavón, N., Ferruz, J. (2007). Taller de Programación Concurrente: Diseño de software para robots basado en hilos Posix. *XXVIII Jornadas de Automática*, Huelva.

Pavón, N., Ferruz, J. (2009). B.EN.DE.R. 2.0: Basic ENvironment for DEveloping Robotic software: Application to educational purposes. *5th IEEE International Conference on Mechatronics*. Málaga, Spain, April 14-17.

Xin-qing, Yan, Wen-feng, Li, Ding-fang, Chen (2006) A New Mechanism for Robots Control Based on Player/Stage. *IEEE International Conference on Robotics and Biomimetics*, pp.750-754, Kunming.