Player como Herramienta de Rápida Integración para Robots Móviles Comerciales, en Redes de Laboratorios para la Experimentación en Robótica.

Andrés Felipe Hurtado
Docente investigador
Institución Universitaria Tecnológica de Comfacauca
email: ahurtado@unicomfacauca.edu.co
Sede Santander de Quilichao - Cauca

Juan Sebastián Bravo
Docente investigador
Institución Universitaria Tecnológica de Comfacauca
email: sbravo@unicomfacauca.edu.co
Sede Santander de Quilichao - Cauca

Resumen: en el siguiente artículo se presentan los aspectos más relevantes del proceso de integración a una red de laboratorios para la experimentación en robótica, empleando la red de alta velocidad RENATA; proceso frente al cual se hace uso de un framework, suministrado por el proyecto Player para la operación del robot Corobot.

Para el desarrollo de la interfaz de operación se utiliza el marco de referencia proporcionado por el proyecto "Framework para el desarrollo de laboratorios de acceso remoto sobre redes de alta velocidad en el área de la robótica".

Palabras clave: robótica móvil, proyecto Player, laboratorio remoto, RENATA.

INTRODUCCIÓN

La robótica en el aula demanda recursos mínimos para poder impartir una enseñanza coherente en relación al conocimiento teórico y la práctica asociada al mismo. En este sentido, se presenta una dificultad en los centros de formación, la cual es contar con robots robustos que permitan realizar estudios alrededor de la robótica móvil y la robótica de manipuladores. Los costos de estas plataformas reducen, a unos cuantas, la posibilidad de tener un aprendizaje empleando estas herramientas.

Una estrategia usada en la actualidad, aprovechando el auge de Internet y las redes de alta velocidad, es operar de forma remota los robots y compartir el recurso entre universidades [1]. Iniciativa que motiva a los centros de formación a unirse en comunidades de formación virtual en lo que respecta a laboratorios de robótica como es el caso de la Universidad del Valle, la Universidad del Quindío y la Institución Universitaria Tecnológica de Comfacauca, quienes conforman la Red de Laboratorios para la Experimentación en Robótica, empleando como canal de comunicación la Red de Alta Velocidad RENATA [2]. Estas redes demandan marcos de referencia en su diseño para que la integración sea rápida y confiable, de tal forma que el mayor recurso de programación sea destinado a complementar los laboratorios y ampliar la variedad de prácticas, aprovechando la particularidad de cada robot [3].

Player surge como una herramienta apropiada para poder integrar robots comerciales de forma rápida a la

red de laboratorios, pues permite controlar todos los actuadores del robot y obtener información sensorial de forma transparente. Player actúa como un servidor que permite controlar al robot y los algoritmos de control son clientes que acceden al robot a través de mensajes, los cuales siguen el protocolo de comunicación de Player para usar las librerías. Una gran ventaja de Player es que posee librerías para ser usadas en diferentes lenguajes como C++, Java, Phyton o Lisp [4].

PROYECTO PLAYER

Player fue desarrollado en sus inicios en la Universidad del Sur de California, en los laboratorios de investigación de robótica, con el propósito de poder acceder de forma eficiente a los actuadores y sensores de los robots disponibles. La idea de tener un servidor para acceder al robot a través de sockets está alrededor de tres motivaciones [5]:

Distribución: un cliente accede a los sensores y los actuadores del robot en cualquier parte de la red.

Independencia: los clientes de Player pueden ser desarrollados en cualquier lenguaje de programación.

Conveniencia: el servidor coloca a disposición de los clientes una interfaz unificada para manejar los dispositivos asociados, el cliente sólo debe suscribir los dispositivos a los cuales desea acceder y definir la frecuencia con la cual desea obtener la información.

En conjunto, el proyecto Player se percibe como un framework para desarrollo de software de control aplicado a robótica, que dispone de amplio soporte para el manejo de dispositivos empleados en laboratorios como webcams, laser, cámaras IP, GPS, etc. Una implementación genérica se puede representar en la figura 1, aunque a simple vista se evidencia la participación de diferentes elementos, por lo cual se parte de una situación bastante compleja a la cual Player responde sin problemas. Pero se pueden realizar aplicaciones más sencillas restando elementos en el diagrama y jugando con las diferentes combinaciones, las que representan situaciones cotidianas en la mayoría de los laboratorios de robótica en las universidades.

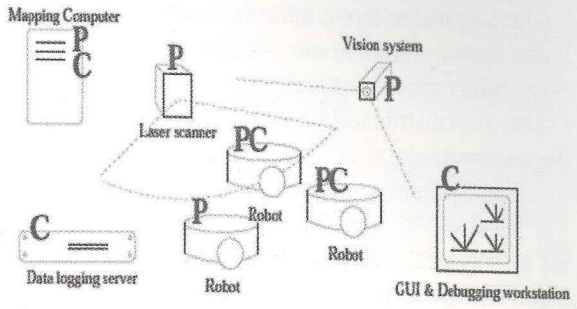

Figura 1. Servidor Player (P), datos de sensores distribuidos para los clientes (C) empleando conexiones por cable o inalámbricas para realizar la comunicación. Figura tomada de Publicaciones del Proyecto Player. Disponible en: http://playerstage.sourceforge.net/index.php?src=pubs (Fecha de actualización: 15 de febrero de 2011).

La relación entre las diferentes capas de abstracción, de la estructura de Player, muestra la forma en que se accede a los dispositivos y el orden en que se desarrolla la implementación en software, lo que evidencia el framework. En la figura 2 se presenta la relación entre las diferentes capas para manejar una cámara web a través de Player.

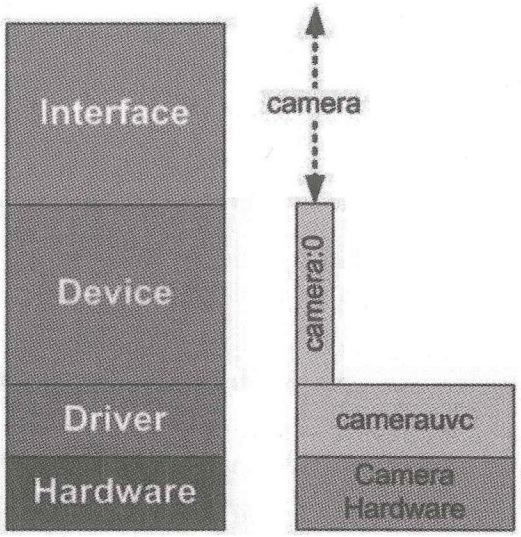

Figura 2. Relación entre el hardware de la webcam y la interfaz del driver que se implementa para acceder a los servicios del dispositivo. Figura tomada de Linux Journal en la publicación "Playing with the Player Proyect". Disponible en: http://www.linuxjournal.com/magazine/playing-player-project (Fecha de actualización: 17 de febrero de 2011).

En la primera capa se encuentra el dispositivo como tal, que para el ejemplo de la figura 2 es una cámara web, pero en esta capa están representados elementos tan comunes en robótica como los sensores entre los

cuales se pueden tener: infrarrojos, sonares, bumpers, entre otros. También se incluyen los actuadores, que comúnmente son los servomotores que se encuentran distribuidos en las llantas y articulaciones, dependiendo de si se trata de un robot móvil o un robot manipulador.

En el segundo nivel se encuentra el driver, en esta capa es donde se ubican los códigos creados, por los programadores, para poder tener el control del respectivo dispositivo. Para el ejemplo el driver de la cámara web se denomina "camerauvc". Una gran ventaja que posee el proyecto Player es la disponibilidad de un numeroso conjunto de drivers para los elementos más comunes en un laboratorio de robótica, conocida como la librería de Player [6].

En el nivel de driver también puede ocurrir que el robot que se desea controlar no se encuentre implementado en las librerías de Player, en ese caso se puede crear el driver para la plataforma, el cual en el proyecto Player recibe el nombre de plugin.

El tercer nivel permite identificar plenamente los elementos del robot, en casos en donde se pueden tener, en una misma plataforma, dos cámaras y bumpers adelante y atrás; entonces esta diferenciación es importante para acceder independientemente a cada dispositivo.

La capa final es lo que se conoce más comúnmente como las API's, que son paquetes a través de los cuales se puede acceder a información del dispositivo y realizar la configuración de sus parámetros [7].

REDES DE LABORATORIOS PARA LA EXPERIMENTACIÓN EN ROBÓTICA.

En la actualidad, la conectividad presente en las universidades de Sur América permite pensar en la creación de redes, para compartir recursos e infraestructuras en el área de robótica, lo cual no podría establecerse de manera individual en cada universidad, debido a los altos costos que estos representan para los laboratorios. Así, se pueden tener muchos escenarios, desde la universidad que ha realizado una gran inversión, en su laboratorio de robótica, hasta la que apenas inicia y desea tener equipos que le permitan realizar proyectos relacionados con esta disciplina.

Inicialmente, para compartir recursos en una red se tienen dos opciones muy comunes, una es a través del internet comercial, la segunda es usar las redes académicas [8]. Estas últimas son la clave para generar una cultura investigativa alrededor del trabajo colaborativo, pues este tipo de redes soportan solamente actividades relacionadas con la investigación y el desarrollo. El recurso es muy potente, ya que está integrado por redes totalmente independientes (tanto en la parte física como lógica).

La primera red de laboratorios de robótica en Colombia, que usó la Red de Alta Velocidad RENATA, fue desarrollada en el marco del proyecto: "Laboratorio distribuido con acceso remoto a través de RENATA para la experimentación en robótica" [9]. Esta red conecta el recurso disponible en los laboratorios de robótica de la Universidad del Valle y la Universidad del Quindío, compartiendo un robot móvil tipo Pioneer 3DX y un robot manipulador Mitsubishi RV-2J.

Este proyecto abre el camino hacia lo planteado al comienzo del capítulo, disponer de una red que permita, a diferentes universidades, beneficiarse de este desarrollo y hacer parte de esta comunidad, la cual se complementa y crece a partir del trabajo realizado. De esta forma, se plantea una segunda iniciativa que permite tener un crecimiento ordenado y una rápida integración a la red de laboratorios planteado en el marco del proyecto: "Framework para el desarrollo de laboratorios de acceso remoto sobre redes de alta velocidad RENATA en el área de la robótica" [10].

La red de laboratorios conformada, en un comienzo, por la Universidad del Valle y la Universidad del Quindío se complementa claramente al permitir a los usuarios realizar prácticas de forma remota en el área de los manipuladores y la robótica móvil. Cada robot dispone de una interfaz de usuario que permite realizar la respectiva manipulación, empleando cámaras IP's, para tener control visual de la operación de las plataformas.

Con el desarrollo del framework otras universidades pueden integrarse de forma rápida, ya que se cuenta con un marco de referencia para realizar esta tarea. Lo que a las universidades creadoras de la red les tomo aproximadamente 2 años. Un nuevo integrante puede tardar aproximadamente 6 meses para ser parte de la red de laboratorios para experimentación en robótica.

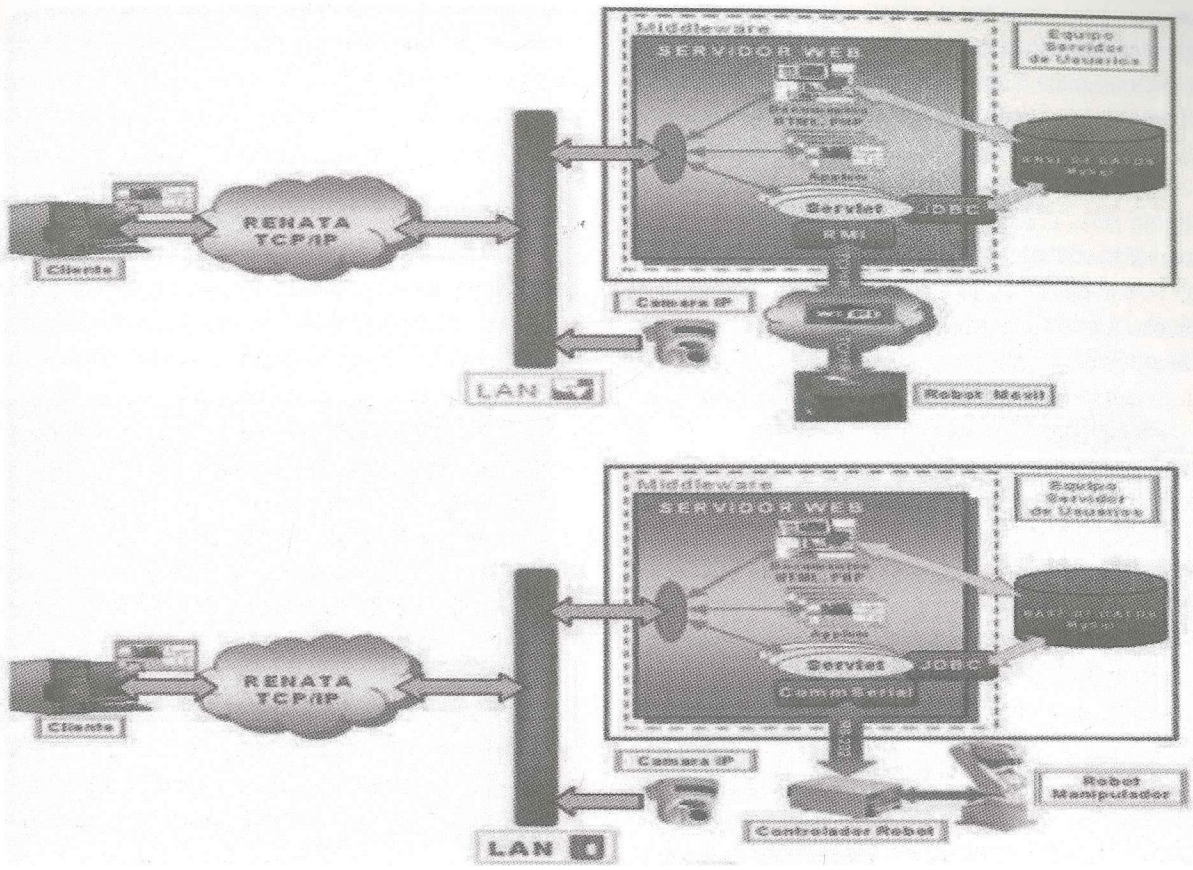

Figura 3. Arquitectura de los laboratorios. Figura tomada de la publicación, "Laboratorio distribuido con acceso remoto a través de RENATA para la experimentación en robótica", disponible en: http://www.renata.edu.co/index.php/descargas/doc_details/52-laboratorio-distribuido-con-acceso-remoto-a-traves-de-renata-para-la-experimentacion-en-robotica.html (Fecha de actualización: 25 de enero de 2011).

En esta parte toma relevancia el proyecto Player, pues el tiempo que se gana al tener un framework (que guía el desarrollo de la aplicación para compartir una plataforma teniendo arquitecturas, como las que se muestran en la figura 3) debe estar acorde con el tiempo en que se realice el código para poder controlar el robot.

Actualmente, un gran porcentaje de los robots comerciales, que poseen control basado en pc's, disponen de las respectivas APIS para realizar el control usando lenguajes de programación como C++ o C#. De igual forma, tienen soporte a través del proyecto Player, es decir, forman parte de las librerías de éste. Es en este punto donde se debe elegir, teniendo en cuenta las características mencionadas del proyecto, el que sea de código abierto y ya que el framework está desarrollado en java, es conveniente privilegiar la colaboración y unir los dos frameworks, en donde se evidencia una reducción amplia del tiempo de integración a la red [11].

COROBOT

El robot móvil corobot es un robot con tracción en las 4 ruedas, con movimiento síncrono, dispone de dos bumpers y una cámara de 2 mega pixeles en la parte delantera.

Asimismo, cuenta con un brazo de 4 grados de libertad, que permite combinar la estrategia de navegación de la robótica móvil con el control en manipuladores. Corobot se caracteriza por su versatilidad a la hora de programarlo, pues cuenta con una board mini ITX Jetway con procesador embebido de 1.5 Ghz y S.O. Windows Xp SP2 y Ubuntu 9.04. Su sistema de percepción se complementa con 2 encoders, conectados a las ruedas delanteras y un sensor de

presión para detectar el cierre de la pinza ubicada en el brazo. Corobot permite ampliar su capacidad sensorial, pues dispone de una tarjeta Phidget 8/8/8 (8 entradas analógicas, 8 entradas digitales y 8 salidas digitales). Posee una velocidad máxima de 45,72 cm/sec y de comunicación inalámbrica a través de una tarjeta pcmcia Wifi. La operación del robot se realiza mediante una interfaz implementada en Java, ubicada en el servidor de usuario y la cual interactúa con el player que corre en el robot bajo el sistema operativo Ubuntu 9.04.

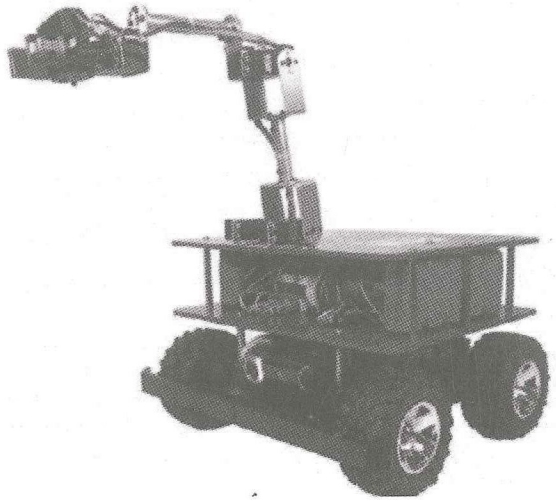

Figura 4. Robot móvil Corobot distribuido por la empresa Coroware

Por último, dentro de las características de Corobot se puede observar la participación de los dos frameworks. Por un lado esta Player, que permite acceder a los actuadores y sensores de Corobot y de otro lado está la interfaz, que permite al usuario realizar la tele operación de la plataforma, la cual se desarrolla de acuerdo a los lineamientos del framework.

COROBOT Y PLAYER

En esta sección se evidencian todos los elementos relacionados con player y lo sencillo que puede ser controlar un robot comercial a través de este framework.

Dentro de los requisitos de instalación es necesario contar con un sistema operativo Linux, lo cual no es problema para un robot con un control basado en pc. Corobot tiene corriendo Windows XP SP2 y Ubuntu 9.04, lo que permite usar las APIS que entrega Coroware y controlarlo realizando aplicaciones que, por ejemplo, corran en C++ y usando el sistema operativo Win XP o instalando player, logrando controlar el robot mediante las librerías que vienen incluidas, realizando la operación de éste bajo el sistema operativo Ubuntu 9.04.

Corobot tiene instalada la distribución de Player versión 3.0.1. Una vez instalado Player se puede probar su ejecución en la plataforma, ejecutando la utilidad playerv, la cual permite interactuar de forma gráfica con todas las interfaces que tiene habilitadas el robot.

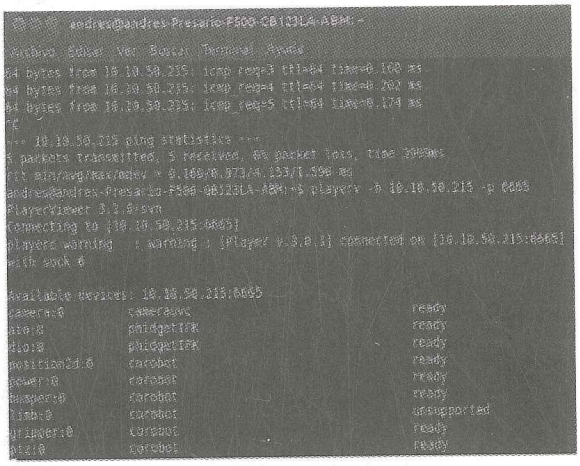

Figura 5. Ejecución de playerv en Corobot

En la figura 5 se observa la información entregada en consola al ejecutar playerv, la estructura de la instrucción es la siguiente:

playerv –h hostname –p port

Se puede observar, en la información de la consola, que se está accediendo a corobot desde un equipo que tiene instalada una distribución de player en versión 3.1.0, la cual está invocando la instrucción playerv y se conecta mediante el socket 6 a Corobot, el cual tiene corriendo una distribución de player en su versión 3.0.1.

El puerto que emplea Player es el 6665 y se muestran todos los dispositivos a los cuales se tiene acceso. Para el brazo de Corobot no se tiene desarrollado el driver por ello aparece como no soportado, éste es el ejemplo perfecto en donde se debe diseñar el driver de acuerdo a los lineamientos del framework para poder controlarlo.

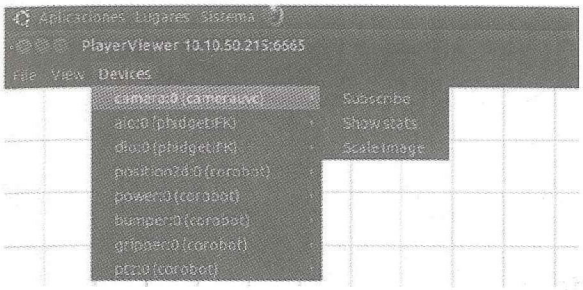

Figura 6. Ventana de playerv

En la ventana de playerv sólo aparecen los dispositivos que están soportados. En la figura 6 se puede observar la forma en que se logra acceder fácilmente a la cámara usb que posee Corobot. Al tomar la opción *subscribe* se pide información de la cámara y se obtiene la imagen de forma inmediata. Por lo tanto, es más clara la abstracción de capas presentada en la figura 2 al usar playerv.

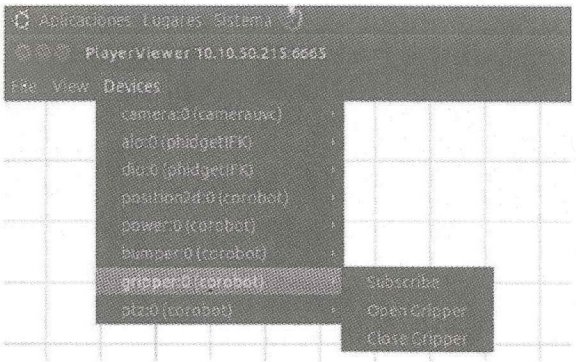

Figura 7. Manipulando un actuador empleando playerv

La pinza que posee Corobot puede ser abierta y cerrada fácilmente, primero suscribiéndose al dispositivo y ejecutando las respectivas operaciones que en este caso son abrir y cerrar. De igual forma, para lograr que el robot se desplace en cualquier dirección se usa position2d.

TRABAJOS FUTUROS

Actualmente los autores están adicionando nuevos elementos a la interfaz de usuario, teniendo en cuenta los lineamientos del framework desarrollado por la Universidad del Valle y la Universidad del Quindío.

Con respecto a los drivers de Corobot, está propuesto el desarrollo del driver del brazo y al complementar la plataforma, en cuanto al sistema de percepción, se tiene proyectado incluir un cinturón de sonares, para lo cual se diseñarán los drivers de estos nuevos elementos.

CONCLUSIONES

Los laboratorios virtuales son una apuesta relevante en la educación moderna, ya que colocan al alcance de todos herramientas y recursos, de alto costo, para fortalecer el aprendizaje de disciplinas como la robótica.

Emplear redes de alta velocidad como RENATA, para la construcción de redes de cooperación frente la enseñanza de la robótica, garantiza el crecimiento de estas iniciativas y lleva a que los investigadores construyan a partir de la experiencia de sus pares.

La mejor forma de garantizar el crecimiento de las redes de laboratorios, para la experimentación en robótica, es contar con frameworks robustos que permitan reutilizar código y disminuyan el tiempo de inserción de nuevos laboratorios a la red.

El proyecto Player es un framework que se acopla a la filosofía de los laboratorios operados de forma remota, posibilitando a los desarrolladores emplear recursos de diseño y programación en las capas superiores, obteniendo así un control del hardware de la plataforma de forma transparente.

REFERENCIAS

[1] F. A. Candelas, S. T. Puente, F. Torres, F. G. Ortiz, P. Gil y J. Pomares. *A Virtual Laboratory for Teaching Robotics.* International Journal of Engineering Education, 2003, Vol. 19, pp. 363-370.

[2] Red de Laboratorios para La experimentación en Robótica, Universidad del Valle, Universidad del Quindío, Institución Universitaria Tecnológica de Comfacauca. 2011. Disponible en: http://200.21.98.247/robotica/

[3] F. A. Candelas, S. T. Puente, F. Torres, V. Segarra y J. Navarrete. *Flexible System for Simulating and Tele-Operating Robots Through the Internet. Journal of Robotic Systems.* 2005, Vol. 22, pp 157-166.

[4] B. G. Gerkey, R. T. Vaughan, K. Stoy, *Player robot server. Technical Report IRIS-00-392*, Institute for Robotics and Intelligent Systems, School of Engineering, University of Southern California, 2000.

[5] B. G. Gerkey, R. T. Vaughan, K. Stoy, A. Howard, G. S. Sukhatme, y M. J. Mataric. *Most Valuable Player: A Robot Device Server for Distributed Control.* 2001. Disponilbe en: http://robotics.stanford.edu/~gerkey/research/final_papers/iros01-player.pdf

[6] B. G. Gerkey, R. T. Vaughan, K. Stoy. *Player Version 1,5.* Publicaciones del Proyecto Player, 2004, Disponible en: http://playerstage.sourceforge.net/index.php?src=pubs

[7] K. Sikorski. *Playing with the Player Project.* Publicacion en Linux Journal, 2009, Vol 188, Disponible en: http://www.linuxjournal.com/magazine/playing-player-project

[8] L. Payá, A. Gil, O. Reinoso, L.M. Jiménez y M. Ballesta. *Plataforma Distribuida para la Realización de Prácticas de Robótica Móvil a través de Internet.* XXVII Jornadas de Automática, 2006, pp. 547-554.

[9] B. Calvache, J. Buitrago, J. Cardona, B. C. Bacca y E. Caicedo. *Laboratorio Distribuido con Acceso Remoto a través de RENATA para la Experimentación en Robótica.* 2009, Disponible en: http://www.renata.edu.co/index.php/descargas/doc_details/52-laboratorio-distribuido-con-acceso-remoto-a-traves-de-renata-para-la-experimentacion-en-robotica.html

[10] B. Calvache, J. Buitrago, E. Caicedo y A. Díaz. *Framework para el Desarrollo de Laboratorios de Acceso Remoto sobre Redes de Alta Velocidad RENATA en el Área de la Robótica.* 2010, Disponible en: http://www.renata.edu.co/index.php/edelectronica-telecomunicaciones-e-informatica/1310-framework-para-el-desarrollo-de-laboratorios-de-acceso-remoto-sobre-redes-de-alta-velocidad-renata-en-el-area-de-la-robotica.html

[11] W. Christian y M. Belloni. *Developing Open Source Programs for science and mathematics.* EUROCON, 2003.