UNIVERSIDAD TECNOLÓGICA EQUINOCCIAL

FACULTAD DE CIENCIAS DE LA INGENIERÍA E INDUSTRIAS

CARRERA DE INGENIERÍA MECATRÓNICA

DESARROLLO DE LA INTERFAZ WEB PARA EL CONTROL Y COMUNICACIÓN DE UN ROBOT HUMANOIDE

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO

DE INGENIERO EN MECATRÓNICA

CARLOS ANDRÉS NOVOA CHICO

DIRECTOR: FAUSTO FREIRE PhD.

Quito, agosto 2016

FORMULARIO DE REGISTRO BIBLIOGRÁFICO

PROYECTO DE TITULACIÓN

DATOS DE CONTACTO	
CÉDULA DE IDENTIDAD:	**1004019111**
APELLIDO Y NOMBRES:	**NOVOA CHICO CARLOS ANDRES**
DIRECCIÓN:	**ENRIQUE RITTER N24-159 Y AV. GASCA**
EMAIL:	**cali_andre@hotmail.com**
TELÉFONO FIJO:	**022505853**
TELÉFONO MOVIL:	**0988880072**

DATOS DE LA OBRA	
TITULO:	**DESARROLLO DE LA INTERFAZ WEB PARA EL CONTROL Y COMUNICACIÓN DE UN ROBOT HUMANOIDE**
AUTOR O AUTORES:	**CARLOS ANDRÉS NOVOA CHICO**
FECHA DE ENTREGA DEL PROYECTO DE TITULACIÓN:	**30 DE AGOSTO DE 2016**
DIRECTOR DEL PROYECTO DE TITULACIÓN:	**FAUSTO FREIRE PhD.**
PROGRAMA	**PREGRADO** [x] **POSGRADO** [ ]
TITULO POR EL QUE OPTA:	**INGENIERO EN MECATRÓNICA**
RESUMEN: Mínimo 250 palabras	En la Universidad Tecnológica Equinoccial como parte de su ejercicio se llevan a cabo distintos proyectos de investigación; en este sentido, dentro de la Facultad de Ciencias de la Ingeniería e Industrias y el Centro de Investigación de Mecatrónica (CIMETICS), se encuentra la implementación de un sistema de teleoperacion para un robot humanoide con el objeto de

realizar pruebas de investigación en el área de robótica y procesos mecatrónicos. Este trabajo de titulación consiste en el desarrollo de un sistema de comunicación inalámbrica para que un robot humanoide pueda ser teleoperado a través de una aplicación web. Para el alcance de este objetivo se emplea la metodología en V de software; así, inicialmente se realizó un análisis de las tecnologías disponibles y los tipos de lenguajes de programación, en virtud de dicho análisis se seleccionó como controlador y servidor web al Raspberry Pi, debido a la multifuncionalidad que puede realizar esta placa electrónica y a Java como lenguaje de programación de la aplicación. Se diseñó el sistema de comunicación entre el Raspberry Pi y el controlador del robot humanoide, para el cual se optó por una arquitectura servidor-cliente para la comunicación de la interfaz gráfica con el servidor, se desarrolló una aplicación web en la cual el operador puede supervisar y controlar las acciones del robot humanoide, para lo cual el usuario tiene dos modos de operación, que son el de control manual y el de control automático.

También se integró una retroalimentación visual a través de una cámara integrada en la cabeza del robot humanoide; en la que respecta al servidor, se montó un serverweb que al ser consumido por las órdenes del operador envía las órdenes al robot humanoide. Mediante pruebas de control se determinó el tiempo de respuesta del robot después de recibir una orden del operador, comparando si el tiempo aumenta o disminuye dependiendo de la ubicación geográfica del operador.

PALABRAS CLAVES: Teleoperación, serverweb, robot humanoide

ABSTRACT: In the UTE, as part of its program, there are various research projects within the Faculties of Science the falcuty of Industial engineering and research center of Mechatronics (CIMETICS), there is an integrated system of teleoperation for a humanized robot to perform research tests in the area of Robotics and processes of mechatronic. This work consists in the development of a system of wireless communication for the humanized robot, so it can be tele-operated by a web application. For the

	scope of this methodology, a software is employed; but initially an analysis has been done of the technologies available and the types of programming languages. After this analysis was done as a driver and web server were selected Raspberry Pi. This because of the multifunctionality that it can perform. The electronic board has Java as the programming language of the application. For the Designed communication system between the IP Raspberry and the humanized robot controller, was opted for an architecture client-server for the communication of the graphical interface with the server. A web application is developed in which the operator can monitor and control the actions of the humanized robot, for which the user has two modes of operation, which are the manual control and automatic control. Also integrated as feedback visual through a camera integrated in the head of the robot humanoid; when it comes to server, mounted a serverweb that sends orders to the humanoid robot to be consumed by the orders of the operator. Through tests of control is determined the time of response of

	the robot after receive an order of the operator, comparing if the time increases or decreases depending on the location geographic of the operator.
KEYWORDS	Teleoperation, serverweb, humanoid robot

Se autoriza la publicación de este Proyecto de Titulación en el Repositorio Digital de la Institución.

f:_____

NOVOA CHICO CARLOS ANDRÉS

1004019111

DECLARACIÓN Y AUTORIZACIÓN

Yo, **NOVOA CHICO CARLOS ANDRES**, CI: 1004019111 autor del proyecto: **Desarrollo de la interfaz web para el control y comunicación de un robot humanoide** previo a la obtención del título de **INGENIERO EN MECATRÓNICA** en la Universidad Tecnológica Equinoccial.

1. Declaro tener pleno conocimiento de la obligación que tienen las Instituciones de Educación Superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.
2. Autorizo a la BIBLIOTECA de la Universidad Tecnológica Equinoccial a tener una copia del referido trabajo de graduación con el propósito de generar un Repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Quito, 30 de agosto del 2016

f:_____

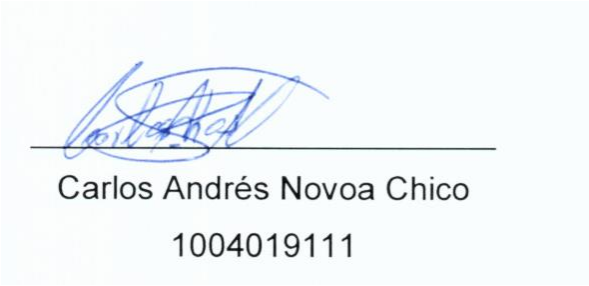
NOVOA CHICO CARLOS ANDRES

1004019111

DECLARACIÓN

Yo **NOVOA CHICO CARLOS ANDRES**, declaro que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

La Universidad Tecnológica Equinoccial puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.



Carlos Andrés Novoa Chico

1004019111

CERTIFICACIÓN

Certifico que el presente trabajo que lleva por título "**Desarrollo de la interfaz web para el control y comunicación de un robot humanoide**", que, para aspirar al título de Ingeniero en Mecatrónica fue desarrollado por **Carlos Novoa**, bajo mi dirección y supervisión, en la Facultad de Ciencias de la Ingeniería e Industrias; y cumple con las condiciones requeridas por el reglamento de Trabajos de Titulación artículos 19, 27 y 28.

Fausto Freire PhD.
DIRECTOR DEL TRABAJO
C.I. 1802424737

DEDICATORÍA

Este trabajo está dedicado a mi familia, por su apoyo brindado durante todos mis años de estudio, en los buenos y malos momentos vividos hasta alcanzar este objetivo.

Principalmente a mis padres Medardo e Inés, que me apoyan en todo lo que necesite y siempre han querido lo mejor para mí. Gracias a ellos he conseguido este logro.

A mis hermanos Rene, Lorena y Mónica por estar siempre pendientes, por querer ayudarme para superarme.

A mis sobrinos, que van a poderme ver graduado.

AGRADECIMIENTO

Agradezco a mi familia por su constante apoyo, cariño y buenos deseos.

A mis padres por ser mis primeros educadores, de estar siempre pendientes para que tome las decisiones correctas, motivarme en los momentos difíciles, disfrutar conmigo las metas y logros conseguidos y haber puesto toda su confianza en mí

A mis amigos y seres queridos por estar a mi lado compartiendo las vivencias y apoyándome durante este proceso.

Y finalmente a la Universidad Tecnológica Equinoccial, mi segundo hogar.

ÍNDICE DE CONTENIDOS

PÁGINA

ÍNDICE DE TABLAS

PÁGINA

ÍNDICE DE FIGURAS

PÁGINA

ÍNDICE DE ANEXOS

PÁGINA

RESUMEN

En la Universidad Tecnológica Equinoccial como parte de su ejercicio se llevan a cabo distintos proyectos de investigación; en este sentido, dentro de la Facultad de Ciencias de la Ingeniería e Industrias y el Centro de Investigación de Mecatrónica (CIMETICS), se encuentra la implementación de un sistema de teleoperacion para un robot humanoide con el objeto de realizar pruebas de investigación en el área de robótica y procesos mecatrónicos. Este trabajo de titulación consiste en el desarrollo de un sistema de comunicación inalámbrica para que un robot humanoide pueda ser teleoperado a través de una aplicación web. Para el alcance de este objetivo se emplea la metodología en V de software; así, inicialmente se realizó un análisis de las tecnologías disponibles y los tipos de lenguajes de programación, en virtud de dicho análisis se seleccionó como controlador y servidor web al Raspberry Pi, debido a la multifuncionalidad que puede realizar esta placa electrónica y a Java como lenguaje de programación de la aplicación. Se diseñó el sistema de comunicación entre el Raspberry Pi y el controlador del robot humanoide, para el cual se optó por una arquitectura servidor-cliente para la comunicación de la interfaz gráfica con el servidor, se desarrolló una aplicación web en la cual el operador puede supervisar y controlar las acciones del robot humanoide, para lo cual el usuario tiene dos modos de operación, que son el de control manual y el de control automático. También se integró una retroalimentación visual a través de una cámara integrada en la cabeza del robot humanoide; en la que respecta al servidor, se montó un serverweb que al ser consumido por las órdenes del operador envía las órdenes al robot humanoide. Mediante pruebas de control se determinó el tiempo de respuesta del robot después de recibir una orden del operador, comparando si el tiempo aumenta o disminuye dependiendo de la ubicación geográfica del operador.

ABSTRACT

In the UTE, as part of its program, there are various research projects within the Faculties of Science the falcuty of Industries engineering and research center of Mechatronics (CIMETICS), there is an integrated system of teleoperation for a humanized robot to perform research tests in the area of Robotics and processes of mechatronic. This work consists in the development of a system of wireless communication for the humanized robot, so it can be tele-operated by a web application. For the scope of this methodology, a software is employed; but initially an analysis has been done of the technologies available and the types of programming languages. After this analysis was done as a driver and web server were selected Raspberry Pi. This because of the multifunctionality that it can perform. The electronic board has Java as the programming language of the application. For the Designed communication system between the IP Raspberry and the humanized robot controller, was opted for an architecture client-server for the communication of the graphical interface with the server. A web application is developed in which the operator can monitor and control the actions of the humanized robot, for which the user has two modes of operation, which are the manual control and automatic control. Also integrated as feedback visual through a camera integrated in the head of the robot humanoid; when it comes to server, mounted a serverweb that sends orders to the humanoid robot to be consumed by the orders of the operator. Through tests of control is determined the time of response of the robot after receive an order of the operator, comparing if the time increases or decreases depending on the location geographic of the operator.

1 INTRODUCCIÓN

En la actualidad existen muchas circunstancias en las cuales no es conveniente emplear personas para la realización de algunas labores debido al alto riesgo al que ellos se exponen; por esta razón, se han desarrollado diversas herramientas o equipos que permiten reemplazar al hombre en la ejecución de estas operaciones a distancia. Dentro de estos dispositivos se encuentran los móviles teleoperados, estos, también son conocidos como robots (a pesar de no ser completamente autónomos). Los robots teleoperados son aquellos controlados por un usuario a distancia desde una estación remota.

En este sentido, existen varias arquitecturas utilizadas para un sistema teleoperado, las más comunes son los que se componen principalmente de una estación de teleoperacion, un sistema de comunicación y esclavo, el esclavo puede ser un robot móvil equipado con un manipulador ubicado en una estación de control. La estación de teleoperación permite controlar al esclavo a distancia por medio del sistema de comunicación; a través de este sistema se pueden transmitir las señales de control hacia el esclavo y, a su vez, recibir señales de información sobre el estado de éste en la estación de control a través de un canal de comunicación que puede ser una red de computadores, un enlace de radiofrecuencia o microondas.

Dentro de la interfaz de teleoperación de robots se manejan varios tipos de información como; trasmisión de video, simulación virtual, comandos de control, datos de sensores en tiempo real, es decir, información que es necesaria procesar bajo ciertas restricciones de tiempo como son tiempo de respuesta, tiempo de retardo y tiempo de procesamiento, las cuales servirán para tomar alguna acción. Para obtener la información necesaria que presenta gráficamente la interfaz sobre los robots móviles estos están provistos de una gran cantidad de sensores, cámaras que se encargan de detectar magnitudes como distancia, tiempo, temperatura, masa, etc. Y a la vez envían dicha información por medio de su tarjeta controladora.

En Ecuador por ejemplo, la teleoperación de robots está siendo desarrollada a través de una iniciativa llamada Teebot. En ésta, se presentan dos empresas que basan su tecnología en software libre, y pretenden iniciar a niños desde los cuatro años de edad en actividades referentes a lógica de programación y robótica. Las empresas EGM Robotics y Clear Minds se fusionaron para desarrollar el robot Teebot., Clear Minds, explica que EGM se encargó de la creación del hardware, mientras que Clear Minds desarrolló el software utilizando software libre. “EGM se ha tardado dos años en desarrollar la placa en la cual el hardware está perfectamente funcional y permite realizar varias actividades adicionales como, por ejemplo, la instalación de sensores, dispositivos bluetooth, luces y dispositivos sonoros, de manera que sea mucho más atractivo para los niños.”

Como objetivo general del proyecto se propuso:

- Desarrollar la interfaz Web que controlará todas las capacidades (caminar hacia el frente, atrás, izquierda y derecha) de un robot humanoide autónomo.

Los objetivos específicos a desarollar son:

- Investigar el lenguaje de programación que presente más beneficios al momento del desarrollo de la aplicación que se encontrará en el servidor Web.
- Determinar los parámetros de control necesarios que debe tener la página Web en su interfaz para la teleoperación del robot humanoide.
- Desarrollar el sistema de adquisición de datos del servidor para el robot humanoide.
- Desarrollar el sistema de comunicación entre el robot humanoide y el servidor de la página Web.

Con la realización de este proyecto se pretende generar la interfaz Web para el control de un robot Humanoide autónomo, por medio del desarrollo de una aplicación que se comunica con el servidor Web. En la interfaz de control se colocarán diferentes tipos de comandos para las acciones a realizar del robot humanoide. Las acciones implementadas son; caminar hacia adelante, atrás,

izquierda y derecha, igualmente serán monitoreadas por una cámara localizada en el robot humanoide con su respectiva señal de video en la interfaz Web. La comunicación entre el servidor y el robot humanoide se la realizará de manera inalámbrica.

Asimismo, este trabajo de titulación realza la importancia de usar herramientas open source de software y hardware para el desarrollo del proyecto, por lo que incentiva el crecimiento de la comunidad tecnológica dedicada al desarrollo de estas herramientas. Así, el eje central del proyecto es la herramienta libre Raspberry pi.

El alcance del proyecto tendrá las siguientes características:

- El sistema de comunicación inalámbrica entre el servidor y la interfaz web funcionará dentro de toda la Universidad Tecnológica Equinoccial mientras esté conectado a la intranet.
- El robot será necesariamente controlado y supervisado por una persona, ya que es un sistema de teleoperacion y depende de las órdenes del operador.
- El sistema de visualización para la interfaz web mostrará el video en tiempo real con resolución mínima de 320 por 360 pixeles.

2 MARCO TEÓRICO

2.1 ROBÓTICA

La robótica es una rama de la ingeniería, que está implicada en el diseño, fabricación y funcionamiento de robots desarrollados para desempeñar tareas realizadas por el ser humano. En esta rama de la ingeniería están comprendidas la electrónica, la informática, la mecánica, la inteligencia artificial, la mecatrónica, la nanotecnología y la bioingeniería.

2.1.1 ROBÓTICA MOVIL

La robótica móvil es el campo de la robótica que se dedica al desarrollo de sistemas de automoción que permiten desplazar al robot de un sitio a otro, estos desplazamientos pueden ser teleoperados por humanos o tener un desplazamiento autónomo.

Los robots móviles están desarrollados para realizar diferentes tipos de tareas para ayudar al ser humano como son: robots para inspección de volcanes donde su misión consiste en determinar composición de gases directamente del cráter, robots espaciales que se dedican a la exploración de otros planetas del sistema solar, aviones no tripulados que se usan para tareas de reconocimiento de lugares, robots desactivadores de bombas que son empleados en operaciones de alto riesgo.

2.1.1.1 Robots humanoides o androides

Los androides son robots humanoides diseñados para ser similares a los seres humanos. Son construidos con la misma estructura física básica y capacidades cinéticas de los seres humanos, aunque no están destinados a tener todas las características de las personas. Pueden tener brazos y piernas articuladas, pueden moverse de la misma manera como lo hacen los humanos, pero tienen un material exterior (plástico o metal) que no imita la apariencia humano (figura 1).

Figura 1. Robot humanoide Asimo
(WikimediaCommons, 2016)

2.1.1.2 Robots teleoperados

Esta clase de robots son controlados de una forma remota, las señales de control remoto se pueden trasmitir a través de un cable, de un sistema inalámbrico local (Wi-Fi), de internet o por satélite. Por definición, toda aplicación con un robot teleoperado requiere de un operador humano, una interfaz de control y el robot teleoperado (figura 2) propiamente dicho (WikiRobotica, 2016).

Figura 2. Robot teleoperado
(Bennet, 2002)

2.1.1.3 Robots Autónomos

Los robots autónomos tienen la capacidad de obtener información sobre sus entornos, y trabajar durante un período prolongado de tiempo sin intervención humana. Estos robots autosuficientes se mueven individualmente durante su funcionamiento sin ayuda humana, y son capaces de evitar situaciones que

son perjudiciales para sí mismos o las personas. Los robots autónomos también presentan algoritmos susceptibles a la adaptación a un entorno cambiante (figura 3).

Figura 3. Robot humanoide Plen
(WikimediaCommons, 2016)

2.2 TELEOPERACIÓN

Teleoperación, también llamado telerobótica, es el término técnico para el control remoto de un robot. En un sistema tele-robótico, un operador humano controla los movimientos de un robot desde cierta distancia. Las señales se envían al robot para controlarlo; otras señales regresan, e indican al operador que el robot ha seguido las instrucciones. Estas señales de control y de retorno se conocen como telemetría. Algunos robots teleoperados tienen un rango limitado de funciones.

Un sistema de teleoperación consta de los siguientes elementos (Alencastre M., 2003) (figura 4):

- Operador o teleoperador: es el humano que manipula las funciones del robot a distancia, este control puede ser continuo o por momentos específicos.
- Dispositivo teleoperado: es la máquina, manipulador o robot que es controlado a distancia por un teleoperador.

- Interfaz: es el conjunto de botones, canales o comandos del sistema operativo, que posee un formato de visualización gráfica para permitir al usuario comunicarse y utilizar el ordenador o programa.
- Control y canales comunicación: son los dispositivos que se encargan de la transmisión y recepción de los datos, señales y órdenes del robot o máquina que se encuentran manipulados.
- Sensores: son los dispositivos físicos que transmiten la información, tanto al robot como al teleoperador, para ser utilizada por el interfaz y el control.

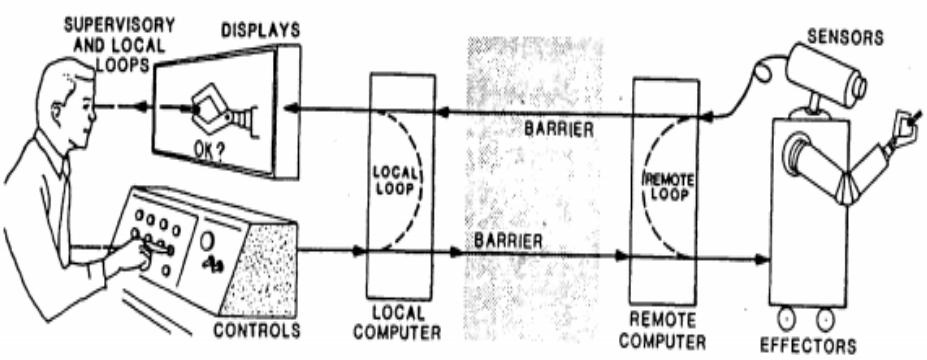


Figura 4. Elementos básicos de un Sistema de teleoperacion
(Alencastre M., 2003)

2.2.1 MÉTODOS DE CONTROL EN LA TELEOPERACIÓN

La teleoperacion a través de sus métodos, en los sistemas de control debe cumplir los siguientes objetivos según (Ollero, 2001):

- Hacer que el control manual del operador humano sea robusto ante retardos, saturación de los actuadores y otras no linealidades, e incluso ante los propios errores del operador.
- Permitir elevadas prestaciones en la teleoperación (precisión, tiempo de teleoperación, comodidad) haciendo que los bucles de control tengan un comportamiento dinámico apropiado y reduciendo el trabajo del operador para lo cual se llega a realizar eventualmente de forma automática la acomodación y el control de esfuerzos en esquemas de control compartido. La reducción del tiempo de teleoperación con el sitio remoto debe ser importante, debido a que la ventana temporal de comunicaciones puede ser limitada.

Un aspecto importante es la determinación de la información que se suministra al operador, en muchos sistemas, la información sensorial fundamentalmente es visual.

2.2.1.1 Control bilateral

Antes debemos mencionar que el control unilateral según (Aracil, 2002) es, cuando el maestro genera las señales de referencia, ya sean de posición o velocidad, para los bucles de control de las articulaciones del esclavo.

Control bilateral: es cuando la realimentación de fuerzas generadas en el robot operado son trasmitidas al operador, por medio de la cinestésica en la que el sistema convierte la fuerza de contacto del esclavo en una fuerza aplicada sobre la mano del operador (Aracil, 2002).

La dificultad con el control bilateral se presenta cuando el esclavo debe seguir los movimientos realizados por el operador, así como la realimentación cinestésica de las fuerzas del esclavo al operador en un sistema de comunicación que podría tener retardos, se han hecho muchos estudios en el caso de Internet para disminuir los problemas ocasionados por dichos retardos (Barnes, 1994)

Una fácil representación de control bilateral es con la arquitectura de (Aracil, 2002) que presenta al maestro y el esclavo con una cinemática proporcional, y el control se realiza articulación a articulación como se observa en la figura 5.

- Control maestro-esclavo

  Esta arquitectura el control se retroalimenta los ángulos para cada articulación, así como los esfuerzos son ejercidos en el esclavo.

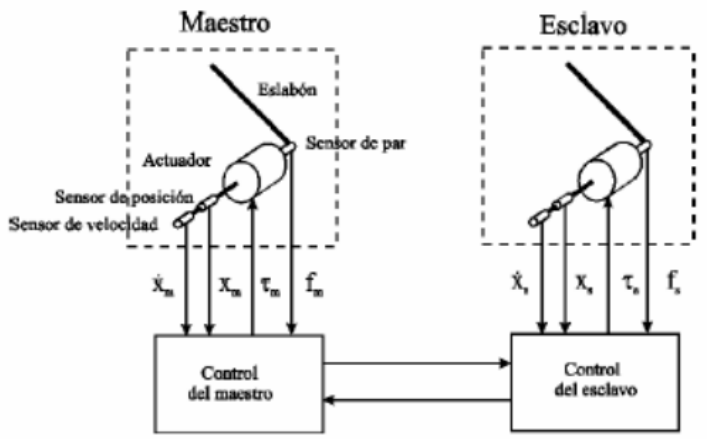


Figura 5. Control mestro-esclavo
(Aracil, 2002)

2.2.1.2 Control supervisado y coordinado

En el control coordinado el operador controla los actuadores, pero está implementado un lazo de control incluido en el sitio donde se encuentra el esclavo, de esta manera no hay autonomía en el esclavo, los lazos cerrados de control en el esclavo son usados cuando el operador no puede controlar directamente el esclavo debido a los retardos en la comunicación. Los sistemas en los que hay un lazo de control digital cerrado utilizan esta estructura de control. Un ejemplo acerca de este tipo de control es (Dialaiti, 2002) el esquema se muestra en la figura 6.

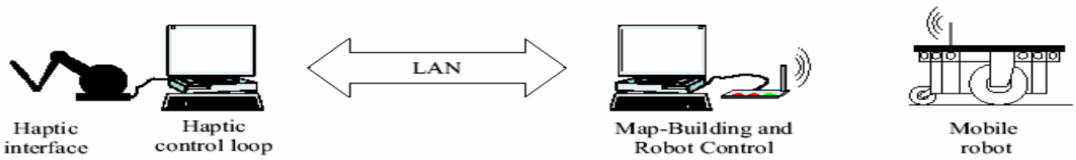


Figura 6. Teleoperación de un robot móvil usando control coordinado
(Dialaiti, 2002)

En el control supervisado el esclavo puede realizar varias tareas semiautónomas, mientras que el operador monitorea y da las órdenes de control al manipulador para que las ejecute, estos sistemas son los más usados y desarrollados en la actualidad, ya que permiten utilizar modelos virtuales y monitorización en tiempo real. Un ejemplo claro de este tipo de control se aprecia en la figura 7.

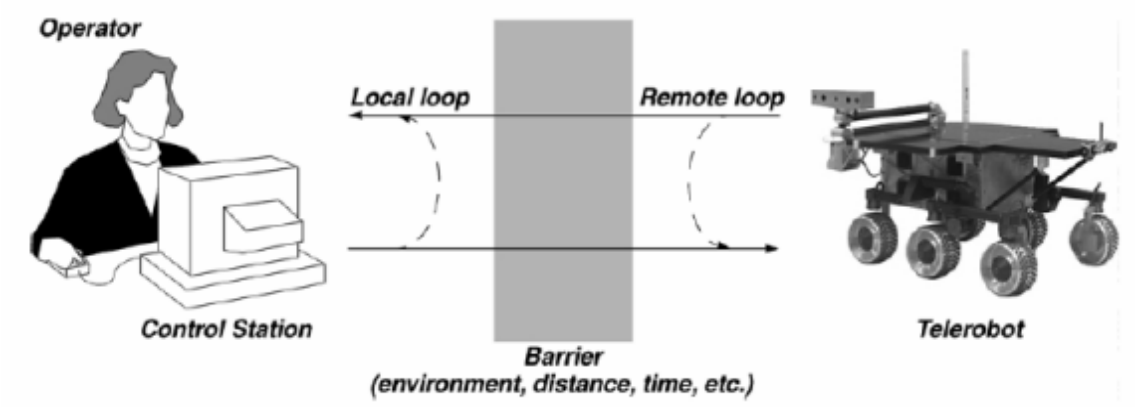


Figura 7. Sistema de control supervisado
(Fong, 2001)

2.2.2 INTERFACES

Las interfaces son el punto de contacto entre un equipo y un usuario, las interfaces según (Fong, 2001) se dividen en tres categorías; Interfaces directas, multimodal o multisensorial y de control supervisado.

2.2.2.1 Directas

Estas interfaces son las más comunes, el operador realiza las órdenes de control para el manipulador o vehículo a través de controladores de mano, como son joysticks, palancas de videojuegos o applets de java en el ordenador, también poseen la realimentación visual por medio de cámaras implementadas en el campo remoto, el operador puede sentir que está en el lugar remoto, este tipo de interfaz es usada cuando el retraso en la trasmisión de órdenes es muy bajo, es decir, en tiempo real (figura 8).

Figura 8. Ejemplo de interfaz directa

(Futuba, 2015)

El primer robot operado desde internet fue creado por el proyecto mercury, la interfaz web se puede observar en la Figura 9. (Golberg & M., 1995).

Figura 9. Interfaz Web directa del primer robot teleoperado
(Telegarden, 1995)

2.2.2.2 Multimodal o multisensorial

Las interfaces multimodales consisten en la presentación de dispositivos y presentación de datos como: auditiva, visual, táctil y gestual conjunta desde cualquier sitio, en cualquier momento, utilizando cualquier dispositivo y de forma accesible, incrementando así la interacción entre personas, así como entre dispositivos y personas.

Las interfaces multisensoriales reciben información de sensores y la representan en un gráfico, estos gráficos ayudan al operador a controlar mejor el robot. En las figuras 10 y 11 se muestran ejemplos de este tipo de interfaces:

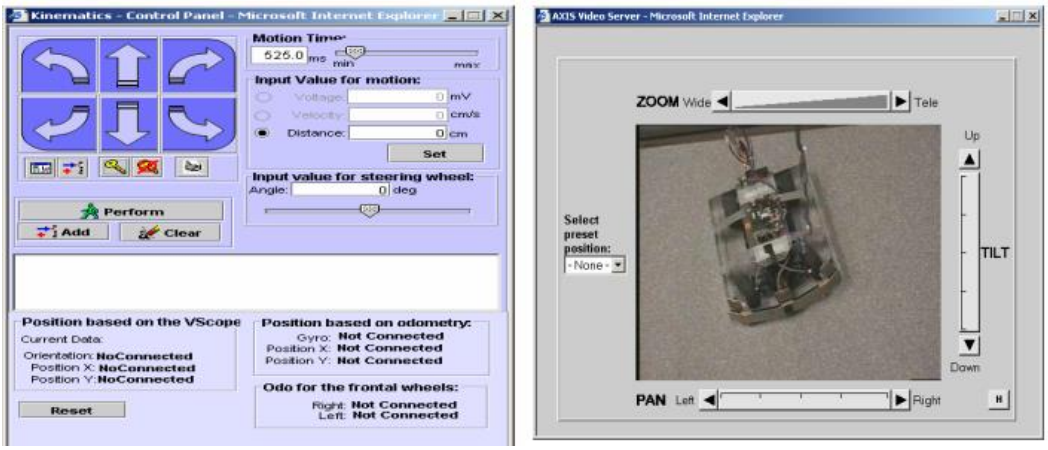


Figura 10. Ejemplo de interfaz multisensorial
(Merlin, 2004)

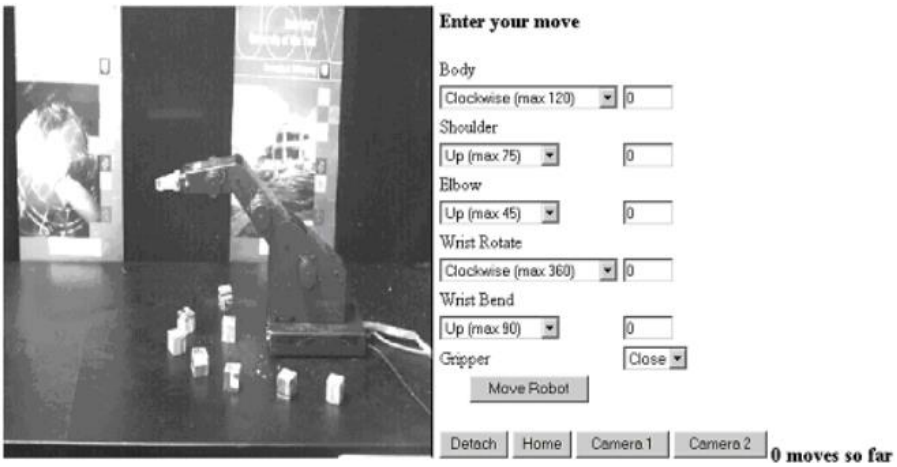


Figura 11. Interfaz multimodal
(Robotoy, 2002)

2.2.2.3 Interfaces para control supervisado

En estas interfaces se intenta recrear todo el entorno al que está sometido el esclavo para maximizar la telepresencia del operador en el sitio remoto, se utilizan dispositivos hápticos con realimentación de fuerza y torque, también se usan dispositivos de realidad virtual, como guantes, lentes, trackers, todo dispositivo tecnológico que permita maximizar la telepresencia del operador.

Figura 12. Dispositivos hápticos
(Century3D, 2016)

2.3 PLACA DE CONTROL

La Raspberry Pi es una pequeña computadora del tamaño aproximado de una tarjeta de crédito; ella cuenta con un procesador, RAM y los puertos de hardware típicos que se encuentran en la mayoría de las computadoras. Esto significa que es capaz de realizar las funciones de una computadora de escritorio, como la edición de documentos, la reproducción de vídeo de alta definición, juegos, codificación y mucho más (figura 13).

Existen muchas funciones que puede realizar la Raspberry Pi, las cuales pueden ejecutarse con equipos de alto costo como correr como una casa NAS (Network Attached Storage), servidor web, centro de medios, TorrentBox y mucho más.

El sistema operativo principal para el Pi es Raspbian; este sistema está basado en Debian, y se trata de una distribución de Linux. No obstante, aunque el sistema operativo compatible principal es Raspbian, se pueden instalar otros sistemas operativos como Ubuntu yegua, Ubuntu Core, OSMC, RIS OS, Windows 10 IO, entre otros.

Figura 13. Raspberry Pi 2 modelo B
(Raspberry, 2016)

2.3.1 SOC (SISTEMA EN CHIP)

El procesador con la que funciona la Raspeberry Pi es un procesador multimedia Broadcom BCM2836 (CPU + GPU + DSP + SDRAM + Puerto USB). Esto significa que todos los componentes presentados, incluidos la central de gráficos, audio y comunicaciones están integradas en el procesador.

- **Arquitectura**

Los componentes principales de procesamiento de datos dentro de la Raspberry Pi basado en SoC son:

- Unidad central de procesamiento (CPU).
- Unidad de Instrucciones RISC de 32 Bits.
- Unidad de procesamiento gráfico (GPU).
- RAM, ROM.
- Controladores del sistema, de memoria, de datos, programadores.
- Chips de gestión de periféricos (USB, microSD).
- Conectividad Ethernet.
- Circuitos de energía, sistema de regulación.

En la figura 14 se encuentra representado el diagrama de un SoC.

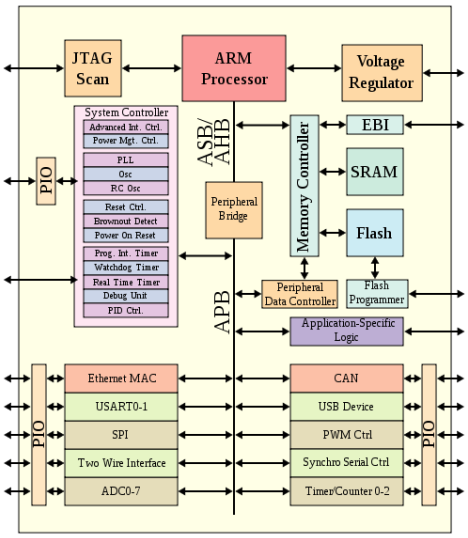


Figura 14. Sistema SoC basado en un microcontrolador
(Burnett., 2007)

2.4 SERVIDORES WEB

Los servidores son ordenadores en los cuales se encuentran programas para gestionar, almacenar o controlar cualquier aplicación realizando conexiones bidireccionales y/o unidireccionales con el cliente. El cliente suele recibir su información a través de la ejecución de un navegador web, la trasmisión de datos suelen utilizar varios protocolos de comunicación, aunque el más usado es el HTTP (figura 15).

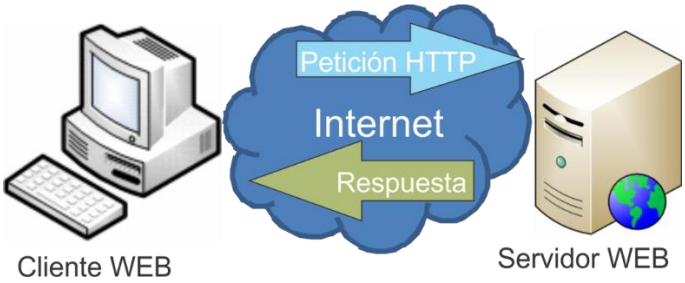


Figura 15. Arquitectura de Servidor Web
(ComputerHope, 2016)

2.4.1 SERVIDOR DE NOMBRE DE DOMINIO

DNS son las siglas de Domain Name System, es decir, Sistema para Nombres de Dominio, que consiste en un sistema jerárquico para dar nombres, más o menos claros, como www.google.com o www.ute.edu.ec a los equipos, servicios y computadores conectados a Internet, evitando que el usuario utilice la dirección IP.

DNS abarca miles de computadores que usan el servicio DNS (también llamado BIND, NAMED o NAMESERVER) que se encarga de asociar (*bind* en inglés) los nombres de dominio con su respectiva dirección IP. Estos servidores DNS actúan como “resolvers” o “resolucionadores” de nombres. Un computador que desea comunicarse, por ejemplo, con Google, envía por el puerto 53 un pedido de resolución de nombre al servidor DNS primario; el servidor DNS le responde con la IP de Google (o le devuelve “error: unresolved domain name”). Desde ese momento en adelante, el computador guarda la IP de www.google.com en un caché, utiliza la dirección IP en todo el tráfico subsiguiente (Ortega, 2009).

2.4.2 SERVIDOR WEB HTTP

HTTP son las siglas de HyperText Transfer Protocol, que significa Protocolo de Transferencia de Híper Texto. Es el servicio más notorio en la Internet, ya que la gran mayoría de usuarios está familiarizado con las siglas http:// que se antepone al nombre de un servidor. El programa que se encarga de atender el protocolo http a través del puerto 80 es el programa Apache. Un nombre muy original pero que se ha hecho famoso, ya que Apache es el nombre del servidor http más famoso y más utilizado a nivel mundial. Existen versiones del servidor Apache para casi todos los sistemas operativos conocidos, incluyendo Linux, Windows, Solaris, AIX, HP/UX, etc. De hecho, más del 70% de los servidores web del mundo corren alguna versión de Apache.

La WWW y el protocolo http fueron creados en el CERN (Laboratorio Europeo de Investigaciones Nucleares) por Tim Berners-Lee y Robert Cailliau en 1989-90. Esta fue la causa de la gran masificación de Internet, ya que con un solo clic (en los hipervínculos o hipertexto) se podía ver gran cantidad de información desde cualquier parte del mundo (Ortega, 2009).

Apache no es el único servidor web. Existen otros como IIS (Internet Information Services), Tomcat, Cherokee, LightHttp, JBoss y otros. Los programadores de páginas jsp (Java Servlet Pages) suelen instalar Tomcat, JBoss o alguna variante de Apache con soporte Java.

- **Apache**

Apache es el servidor web más utilizado. Desarrollado y mantenido por la Fundación de Software Apache, Apache es un software de código abierto disponible de forma gratuita. Se ejecuta en aproximadamente el 70% de todos los servidores web en el mundo. Es rápido, fiable y seguro. Puede ser altamente personalizado para satisfacer las necesidades de muchos entornos diferentes mediante el uso de extensiones y módulos. La mayoría de los proveedores de alojamiento WordPress utilizan Apache como su software de servidor web.

- **Tomcat**

Tomcat es un servidor de aplicaciones de la Apache Software Foundation que ejecuta Javaservlets, y hace que las páginas Web que incluyen lenguaje Java se ejecuten con un modelo de "implementación de referencia" de la Java Servlet y las especificaciones de Java Server Page, Tomcat es el resultado de una colaboración abierta de desarrolladores y están disponibles las dos versiones existentes ; binaria y de fuente desde el sitio Web de Apache. Tomcat requiere un tiempo de ejecución en el entorno Java empresarial que se ajuste a JRE 1.1 o posterior.

2.4.3 SERVIDOR DE ARCHIVOS (FTP)

FTP son las siglas de File Transfer Protocol, que significa Protocolo de Transferencia de Archivos. Es un mecanismo para cargar archivos desde un equipo cliente al servidor y viceversa. La transferencia se realiza utilizando el chequeo y la corrección de errores, por lo que, cuando se termina la transferencia, el archivo subido al servidor (upload) o bajado (download) desde el servidor queda en un estado coherente y conocido (Ortega, 2009).

Cuando se trata de permitir que los usuarios suban archivos al servidor, o descarguen archivos voluminosos desde el servidor, el protocolo FTP tiene mejor rendimiento y mejores resultados que el protocolo HTTP. El protocolo HTTP carece del chequeo de errores y tampoco tiene retransmisión, por lo que solo se utiliza para pequeños archivos o para un uso ocasional.

Para un uso frecuente, continuo, eficiente y con corrección de errores, sobre todo en enlaces malos, la recomendación es utilizar el protocolo FTP. El protocolo FTP tiene varias modalidades como: FTP normal, FTP activo, FTP pasivo, Secure FTP (utilizando SSH).

El más conocido es el FTP activo, que utiliza dos puertos: el puerto 21 para la comunicación entre el cliente y el servidor, y el 23 para transferir el archivo. Estos puertos deben ser abiertos por los cortafuegos.
Hay disponibles varios programas Linux que implementan el protocolo FTP. Entre ellos está el FtpPro, el vsFtp, el tFtp, el FileZilla, y otros más.

2.4.4 SERVIDOR DE CORREO ELECTRÓNICO

El correo electrónico es una de las funciones más utilizadas en Internet. Millones de “emails” viajan todos los días por Internet. Muchos de esos mensajes son propaganda, correo basura, mensajes no solicitados, engaños o intentos de fraudes.
En este sentido, el correo electrónico es conformado por varios componentes y protocolos. Linux dispone de varias alternativas para configurar el servicio de correo electrónico. Uno de los más conocidos es el sendmail. Para el buen funcionamiento del correo electrónico, es necesario comprender sus componentes y protocolos.

Componentes necesarios:

- Un servidor de correo con nombre de dominio para recibir y almacenar los correos de sus usuarios.
- Un cliente de correo para escribir (o componer) correos y enviarlos al usuario@servidor. Existen varios clientes de correo en modo texto, en modo gráfico y en modo web.
- Algunos componentes adicionales para autenticar usuarios, crear listas de correo, administrar listas, calendarios, contactos, tareas, groupware, proyectos, etc.

Protocolos necesarios:

- SMTP: Simple Mail Transfer Protocol utiliza el puerto 25 para enviar el correo de un servidor a otro. Este protocolo es exclusivamente para MTA (Mail Transfer Agent), o sea, cuando el correo viaja de un servidor a otro. Este protocolo no se usa para comunicación con el cliente. El paquete más usado es el sendmail. También se usa postfix o qmail.
- POP3 Post Office Protocol Version 3 (Protocolo de Oficina de Correo versión 3) utiliza el puerto 110 para que el cliente se comunique con el servidor. Los mensajes residen en el cliente. Cada vez que éste envía un correo, el servidor lo recibe y lo despacha a otro servidor, sin guardar una copia. Cada vez que el usuario recibe un correo, el servidor lo almacena hasta que el cliente lo recoge de su buzón, y lo borra inmediatamente (a menos que se haya configurado la opción del cliente: Guardar los mensajes en el servidor durante X días). POP3 es adecuado para conexiones dial-up; de esta forma el correo reside en el cliente y puede desconectarse. Clientes de correo POP3 son Outlook, Outlook Express, Evolution, Thunderbird, Eudora.
- IMAP Internet Message Access Protocol (Protocolo de Acceso a Mensajes de Internet) utiliza el puerto 143 para que el cliente se comunique con el servidor. Los mensajes permanecen en el servidor, a diferencia de POP3 que los mensajes permanecen en el cliente. Cada vez que el cliente envía un correo, el servidor lo almacena en la bandeja de salida y lo pasa a la bandeja de enviados. Cada vez que el usuario recibe un correo, el servidor lo almacena en la bandeja de entrada y lo conserva allí para que el usuario lo vea las veces que sea necesario hasta que el propio usuario los borre. De esta forma, el IMAP es adecuado para intranets, para usuarios con un buen ancho de banda y para correo institucional porque el correo se almacena en uno o varios servidores institucionales seguros.

2.4.5 WEB SERVICE

Un web service es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como internet.

Un web service es una función que diferentes servicios o equipos utilizan; es decir, solo se envían parámetros al servidor (lugar donde está alojado el web service) y éste responderá la petición. Permite la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar y abiertos. Las especificaciones son gestionadas por una organización abierta, la W3C, por tanto no hay secretismos por intereses particulares de fabricantes concretos y se garantiza la plena interoperabilidad entre aplicaciones (figura 16).

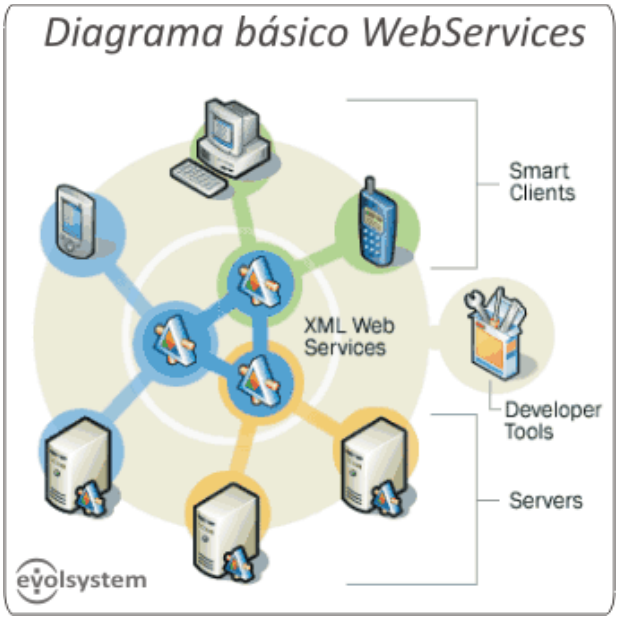


Figura 16 Diagrama básico de WebServices
(Screenvalley, 2016)

2.4.5.1 Webservices tipo rest

REST (Representational State Transfer) es una arquitectura de software para la Web. El término fue introducido en la tesis doctoral de Roy Fielding en 2000, quien es uno de los principales autores de la especificación de HTTP. (Navarro, 2006)

El término se usa para describir a cualquier interfaz que transmite datos específicos de un domino sobre HTTP sin una capa adicional, como hace SOAP. Estos dos significados pueden chocar o incluso fusionarse. Es posible diseñar un sistema software de gran tamaño. Así como también es posible diseñar una simple interfaz XML+HTTP que no sigue los principios REST, y en cambio seguir un modelo RPC (Remote Procedure Calls). Los Servicios Web basados en RPC presentan una interfaz de llamada a procedimientos y funciones distribuidas, la unidad básica de este tipo de servicios es la operación WSDL (WSDL es un descriptor del Servicio Web) (Navarro, 2006).

REST no es un estándar, sino simplemente un estilo de arquitectura. Sin embargo, aunque no es un estándar, está basado en estándares como:

• HTTP
• URL
• Representación de los recursos: XML/HTML/GIF/JPEG/.
• Tipos MIME: text/xml, text/html.

2.4.5.2 Webservices tipo SOAP

SOAP (Simple Object Acess Protocol), es un protocolo de intercambio de información basado en XML, diseñado para Internet. Extiende el protocolo HTTP, y es una herramienta fundamental en los servicios web que permite intercambio de documentos o llamada a procedimientos remotos (RPC). Es independiente de la plataforma y el lenguaje empleado. Es la técnica más purista para implementar un WebService.

Ventajas de SOAP:

- Uso de XML.
- Facilidad para atravesar cortafuegos y proxies al basarse en el modelo get/response de HTTP.

Desventaja de SOAP:

- Los datos binarios se codifican como texto, esto hace que el tiempo de respuesta sea más largo.

Los mensajes SOAP poseen cuatro elementos básicos que son (figura 17).

Envelope (Envoltorio): Define el comienzo y el final del mensaje

Header (Encabezamiento): Atributos para procesamiento del mensaje (seguridad, encriptación, etc).

Body (Cuerpo): Contiene el dato XML a enviar

Fault (Errores): Información sobre errores de procesamiento del mensaje.

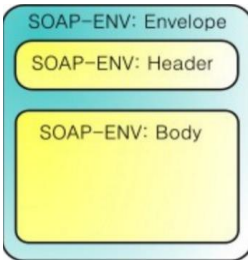


Figura 17. Elementos de un mensaje SOAP
(Java, 2016)

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 299

<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Header>
  </soap:Header>
  <soap:Body>
    <m:GetStockPrice xmlns:m="http://www.example.org/stock">
      <m:StockName>IBM</m:StockName>
    </m:GetStockPrice>
  </soap:Body>
</soap:Envelope>
```

Figura 18. Ejemplo de mensaje SOAP
(Java, 2016)

2.5 TRANSMISIÓN DE VIDEO

La transmisión de video a través de comunicación inalámbrica es una rama muy importante de la comunicación que se estudia para la visualización de imágenes en tiempo real. Este tema se desarrolla muy rápidamente debido al crecimiento masivo que ha supuesto Internet en estos últimos años. A

diferencia del audio, el video ocupa una gran cantidad de ancho de banda los videos en internet se codifican a una compresión de alrededor de 1Mbps.

2.5.1 VIDEO

Video es una combinación de secuencia de imágenes para formar una imagen en movimiento. El video transmite una señal y procesa el orden en el cual se deben mostrar las imágenes capturadas en el monitor.

2.5.1.1 Video digital

El vídeo digital convierte la imagen real captada por la cámara en lenguaje binario de 1 y 0. El vídeo digital es la información del sistema en bits, se puede almacenar en discos duros o en cualquier otro medio de almacenamiento.

Cada imagen del vídeo digital está compuesta de un número concreto de pixeles; éstos son la unidad mínima y los que determinarán la calidad de la imagen digital e igualmente tendrán la información del color. La resolución de la imagen del vídeo digital se mide en pixeles por pulgada o dpi (dots per inch). A mayor resolución mayor calidad de imagen.

Los dispositivos de captura han reducido su tamaño considerablemente en comparación con las cámaras analógicas. La calidad del dispositivo digitalizador resulta primordial para obtener una imagen óptima.

2.5.1.2 Digitalización

La digitalización es el proceso de convertir la información en un formato digital; dentro de este formato, la información se organiza en unidades discretas de datos (llamado bits) que pueden ser tratados por separado (por lo general en grupos de múltiples bits llamados Bytes). Este es el sistema binario de datos que las computadoras y muchos dispositivos con capacidad de procesamiento (tales como cámaras y audífonos digitales) pueden tratar.

La digitalización de la información hace que sea más fácil de conservar, acceder y compartir. Por ejemplo, un documento histórico original solamente

puede ser accesible para las personas que visitan su ubicación física, pero si el contenido del documento se digitaliza, se puede poner a disposición de las personas en todo el mundo. Esto ha generado un impacto importante en la tendencia creciente hacia la digitalización de los datos.

De acuerdo con un artículo publicado en The Guardian en marzo de 2007, si todo el lenguaje hablado desde los inicios del tiempo se digitaliza, se consumiría cinco Exabyte de espacio de almacenamiento. Información digital total, en 2006 se estimó en 161 mil millones de exabytes. El email solo está compuesto por seis exabytes de esa cifra.

2.5.1.3 Bitrate

El bitrate o flujo de datos es la cantidad de información por segundo transmitida por una imagen digital. Esta imagen será de mayor calidad cuanto mayor sea su bitrate.

El video digital no pierde calidad en función del número de reproducciones. Los soportes son más pequeños y manejables que los analógicos por lo almacenarlos es más fácil.

2.5.2 COMPRESIÓN

La técnica de compresión de vídeo consiste de tres pasos fundamentalmente, primero el pre-procesamiento de la fuente de vídeo de entrada, paso en el cual se realiza el filtrado de la señal de entrada para remover componentes no útiles y el ruido que pudiera haber en esta. El segundo paso es la conversión de la señal a un formato intermedio común (CIF), y por último, el paso de la compresión. Las imágenes comprimidas son transmitidas a través de la línea de transmisión digital y se hacen llegar al receptor, donde son reconvertidas al formato común CIF y son desplegadas después de haber pasado por la etapa de post-procesamiento.

Los estándares más conocidos de compresión de video digital son:

- **MJPEG**

  Motion-JPEG es una versión extendida del algoritmo JPEG que comprime imágenes. Básicamente consiste en tratar al vídeo como una secuencia de imágenes estáticas independientes a las que se aplica el proceso de compresión del algoritmo JPEG una y otra vez para cada imagen de la secuencia de vídeo. Existen cuatro modos de operación para el JPEG: secuencial, progresiva, sin pérdida, y jerárquica. Normalmente se utiliza el modo secuencial.

  La ventaja es que se puede realizar en tiempo real e incluso con poca inversión en hardware. El inconveniente de este sistema es que no se puede considerar como un estándar de vídeo pues ni siquiera incluye la señal de audio. Otro problema es que el índice de compresión no es muy grande.

  JPEG utiliza una técnica de compresión espacial, la intracuadros o DCT (formato de cinta de video). El sistema JPEG solamente utiliza la compresión espacial al estar diseñado para comprimir imágenes individuales.

  Motion-JPEG es el método elegido para las aplicaciones donde se envía la misma información a todos los usuarios, las broadcast.

- **AVI**

  (Audio Video Interleaved = Audio y Video Intercalado) Es el formato estándar para almacenar video digital. Cuando se captura video desde una cámara digital al ordenador, se suele almacenar en este formato con el códec DV (Digital Video). El archivo AVI puede contener video con una calidad excelente. Sin embargo, el peso del archivo resulta siempre muy elevado.

  Admite distintos códecs de compresión como CinePak, Intel Indeo 5, DV, etc. Los códecs con más capacidad de compresión y una calidad aceptable son DivX y XviD. El formato AVI puede ser visualizado con la mayoría de reproductores como Windows Media, QuickTime, o VLC.

Siempre y cuando se encuentren instalados en el equipo los adecuados códecs para cada tipo de reproductor.

- **MOV**

  Es el formato de video y audio desarrollado por Apple. Utiliza un códec propio que evoluciona en versiones con bastante rapidez. Este tipo de archivos también pueden tener extensión se recomienda utilizar el reproductor de QuickTime. Existe una versión gratuita del mismo que se puede descargar de Internet, es ideal para publicar videos en Internet por su razonable calidad/peso. Además, admite streaming.

- **WMV**

  Ha sido desarrollado recientemente por Microsoft utiliza el códec MPEG-4 para la compresión de video, también puede tener extensión *.ASF sólo se puede visualizar con una versión actualizada de Windows Media 7 o superior. Esta aplicación viene integrada dentro de Windows es ideal para publicar videos en Internet por razonable calidad/peso.

3 METODOLOGÍA

La metodología en V de software representa el desarrollo por etapas del ciclo de vida de un proyecto. Describe las actividades y resultados que son realizadas durante la duración del desarrollo de la aplicación. La parte izquierda de la “V” representa la descomposición de los requisitos y la creación de las especificaciones del sistema. El lado derecho de la “V” representa la integración de partes y su verificación. “V” significa “Validación y Verificación” (figura 19) (Padilla, 2015).

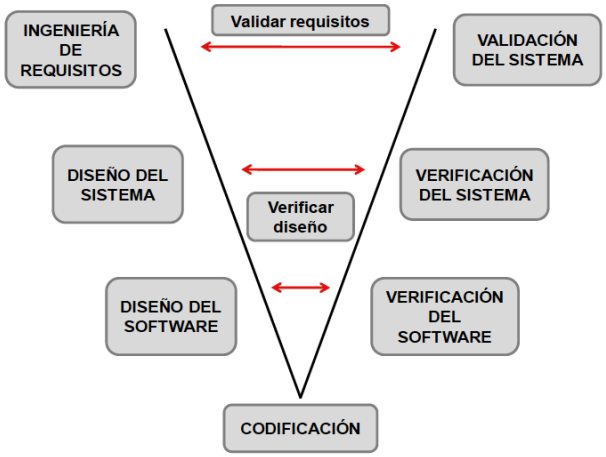


Figura 19. Metodología en V de software
(Padilla, 2015)

La figura 19 presenta la metodología en la cual está basada el desarrollo del presente trabajo de titulación, donde se empezó por las especificaciones y requerimientos del sistema, las cuales dan las directrices necesarias para definir las características del sistema, para proceder paralelamente con el diseño del sistema, diseño de software, verificación y la codificación. Se desarrolla un prototipo para comprobar el funcionamiento de los sistemas; a partir del diseño se realizan las verificaciones por medio de las pruebas necesarias para comprobar su correcto funcionamiento. Este trabajo de titulación trata sobre el desarrollo de una aplicación web para controlar un robot humanoide localizado en la UTE.

El trabajo inicia con la investigación sobre métodos y tecnologías que existen actualmente para el desarrollo de aplicaciones web enfocados a la teleoperación en robots móviles, y a partir de los resultados de la investigación, seleccionar la mejor alternativa y desarrollarla.

Se inició con el diseño del sistema de control que consta de un circuito electrónico de control basado en la placa Raspberry Pi 2, un circuito actuador basado en la placa Arduino nano que entregará la señal de retroalimentación para el sistema de lazo cerrado.

El diseño de control del sistema consiste en una aplicación web en la cual el usuario podrá teleoperar al robot humanoide. La aplicación web cuenta con una virtualización 3D del robot humanoide en la que simula el comportamiento a través de un video animado en función de las órdenes del usuario, también se puede supervisar el desplazamiento del robot en tiempo real gracias a una cámara web conectada al Raspberry Pi.

Finalmente, se ensambla el hardware con el software, con el objeto de realizar las pruebas necesarias al prototipo físico y si existe alguna corrección a realizar, aplicar la correspondiente solución.

3.1 REQUIRIMIENTOS DEL PROYECTO

Para el trabajo de titulación se debe desarrollar una aplicación web que controle al robot humanoide de la UTE, para el diseño de este proyecto se utiliza software y hardware open source, lo que significa el uso de lenguajes y herramientas de programación libres. Usando open source se busca la reducción de costos monetarios para tener mayor facilidad de implementación, la interfaz gráfica de la página web tiene que ser accesible desde la intranet de la UTE, de esta forma el control y monitoreo del desplazamiento del robot humanoide se podrá realizar remotamente desde cualquier lugar y dispositivo conectado a la red de la UTE.

El diseño del proyecto propone los siguientes requerimientos técnicos de ingeniería:

- Control remoto del sistema desde la intranet de la UTE.
- Modo de control automático y manual del robot humanoide.
- Sistema de visualización con retroalimentación de la posición en tiempo real.

- Comunicación con el hardware en tiempo real mediante la intranet.
- Virtualización del robot en la interfaz gráfica.

La tabla 1 presenta un resumen de la función, restricciones y objetivo del trabajo de titulación.

Tabla 1. Requerimientos de diseño del proyecto

Función	Controlar los movimientos del robot humanoide.
Restricciones	Uso de herramientas y código libre.
Objetivos	Desarrollar una interfaz amigable para la fácil manipulación del robot humanoide.

3.2 SELECCIÓN DE ALTERNATIVAS DE DESAROLLO

Para la selección de la mejor alternativa para el desarrollo la aplicación web de control del robot humanoide se deben analizar las tecnologías disponibles y los métodos existentes que permitan el desarrollo técnico del proyecto, aspectos financieros y necesidades del usuario. Para lograr una selección óptima se necesita proponer varias soluciones que puedan cumplir con las funciones del proyecto. La tabla 2 presenta las tres posibles alternativas de solución.

Tabla 2. Alternativas de solución del proyecto

Solución	Alternativa 1	Alternativa 2	Alternativa 3
Dispositivo de control	Rasberry pi 2	Arduino Mega	Microcontroladores PIC
Cámara de video	Cámara Web	Cámara IP	Cámara inalámbrica
Lenguaje de programación	JAVA	PHP	C#
Tecnología de comunicación con el robot	Serial	Wifi	Ethernet
Tecnología de comunicación con la aplicación	WebServer	WebSocket	HTTP
Tecnología de comunicación con la red	Wifi	Ethernet	Radio frecuencia
Software de modelamiento	Solid Works3D	3D-max	Autodesk Inventor
Simulación online	Video /JavaScript	GIF	Flash

Para determinar la alternativa óptima de solución se realiza un análisis mediante el método de criterios ponderados.

3.3 INGENIERÍA DE CRITERIOS PONDERADOS

La ingeniería de criterios ponderados consiste en sopesar una serie de factores que se consideran relevantes para la elección de la mejor alternativa de solución; como son costo, peso, funcionalidad, instalación, mantenimiento, control y vida útil. En función de estos criterios se realiza una calificación con cada alternativa en una escala de 1 al 5 siendo 5 la mejor opción.

En la tabla 3 se presentan los datos con los criterios ponderados para la selección de la mejor alternativa del proyecto.

Tabla 3. Valoración de las alternativas del proyecto

ALTERNATIVAS 5 ES EXCELENTE 1 ES MENOR	**COSTO**	**PESO**	**SERVIDOR WEB**	**INSTALACIÓN**	**MANTENIMEINTO**	**CONTROL**	**VIDA ÚTIL**	**TOTAL**
ALTERNATIVA 1	4	5	5	5	5	5	5	34
ALTERNATIVA 2	4	4	4	4	5	5	4	30
ALTERNATIVA 3	5	5	3	3	3	3	3	25

Una vez ponderadas las alternativas de desarrollo del proyecto, se realiza otra tabla donde se comparan los criterios de ingeniería entre sí para obtener la valoración de cada criterio y así determinar la importancia en el proyecto. Los valores corresponden a la relación que tenga cada criterio con el otro, donde 10 es alta, 5 es media y 0 es ninguna, en la tabla 4 se observan los criterios con mayor relación entre sí.

Para determinar la alternativa ganadora se multiplican los valores de relación entre criterios por los valores de la tabla 5. De esta manera, el resultado del

análisis tiene como ganador a la alternativa 1; esta solución es adecuada para las necesidades de diseño del proyecto, presenta los componentes más importantes en la solución como el servidor web donde se colocará el back-end de la aplicación; el controlador Raspberry Pi 2, que es la placa más funcional en comparación a las otras propuestas ya que servirá como actuador y servidor web simultáneamente; el control de tipo Webserver, éste es el más funcional que se puede implementar ya que estará montado en la Raspberry Pi y podrá ser consumido por cualquier dispositivo que se encuentre en la misma red, así mismo, en comparación los otros propuestos presenta una menor tiempo de retraso en la respuesta del controlador.

Tabla 4. Ponderación de la relación entre criterios con requerimientos de ingeniería

CRITERIOS	COSTO	PESO	SERVIDOR WEB	INSTALACIÓN	MANTENIMEINTO	CONTROL	VIDA ÚTIL	TOTAL	PODNERACIÓN	%
COSTO		0	10	10	10	0	5	35	0.14	13.7%
PESO	0		0	10	0	0	0	15	0.06	5.8%
SERVIDOR WEB	10	0		10	10	10	10	50	0.19	19.4%
INSTALACIÓN	5	10	10		5	0	0	30	0.12	11.7%
MANTENIMIENTO	10	0	10	10		5	10	45	0.18	17.6%
CONTROL	0	0	10	5	10		10	35	0.14	14%
VIDA ÚTIL	10	0	10	10	10	10		50	0.19	19.4%
							SUMA	255	1.00	100%

Tabla 5. Sumatoria de las alternativas en función de criterios ponderados

CRITERIOS	COSTO	PESO	SERVIDOR WEB	INSTALACIÓN	MANTENIMEINTO	CONTROL	VIDA ÚTIL	TOTAL	PONDERACIÓN
ALTERNATIVA 1	0.55	0.3	0.97	0.59	0.88	0.7	0.95	4.94	1
ALTERNATIVA 2	0.55	0.24	0.78	0.47	0.88	0.7	0.76	4.38	2
ALTERNATIVA 3	0.69	0.3	0.59	0.35	0.53	0.42	0.57	3.45	3

4 DISEÑO

4.1 DISEÑO DEL SISTEMA

El sistema se basa en una arquitectura cliente-servidor, donde el componente principal que procesa toda la información y controla es el Raspberry Pi 2. Este dispositivo interpreta las señales de control que recibe tanto por el servidor o por el puerto serial para ejecutar el algoritmo de control y genera las señales correctas para Arduino, que es el que controla el movimiento del robot humanoide, en la figura 20 se presenta el diagrama de funcionamiento del sistema.

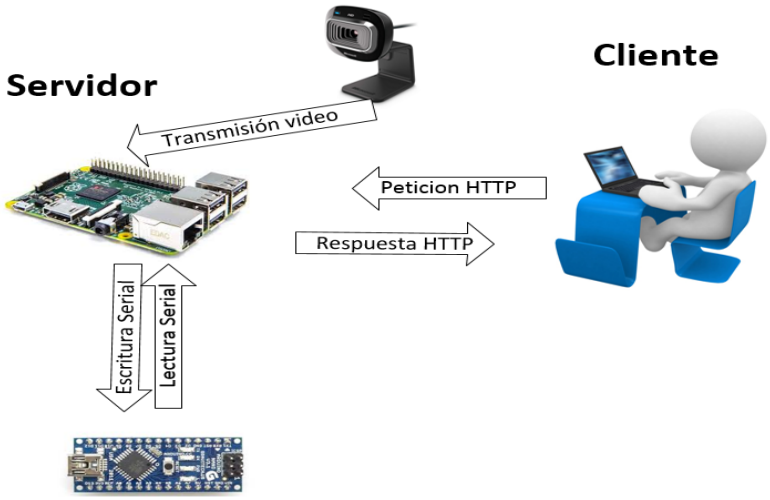


Figura 20. Diagrama de funcionamiento del Sistema

4.1.1 SISTEMA DE CONTROL DEL ROBOT

El sistema de control del robot humanoide permite enviar señales de control desde el operador hasta el servidor web, el servidor recibe las señales a través del consumo restful del server y genera la respectiva señal para el controlador del robot humanoide, el sistema está compuesto de una página HTML con los parámetros de control, funciones de consumo restfull a través de Jquery y el controlador del robot conectado al servidor a través del puerto USB como se lo presenta en figura 21 (Freire, 2013).

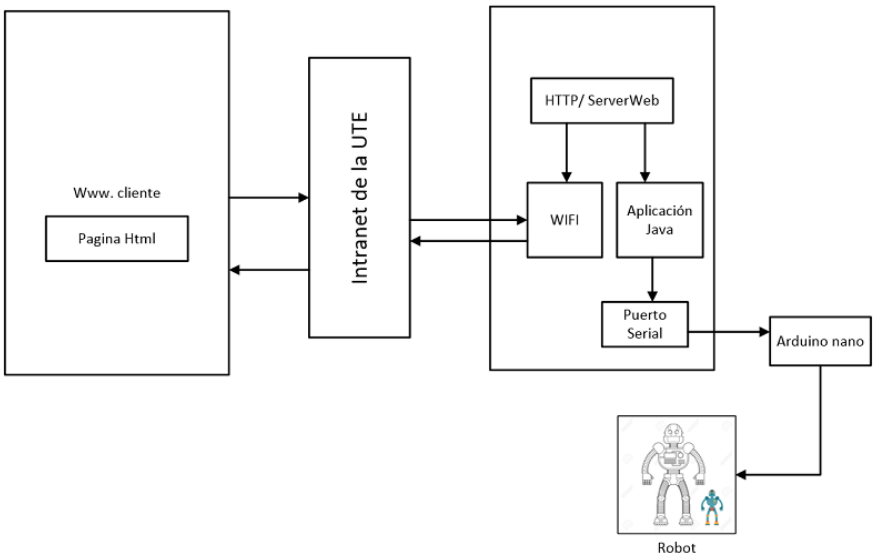


Figura 21. Esquema de Tele-control del robot

Las señales de control son los comandos de dirección para movimiento del robot, posición de la cámara. Las señales de control enviados por el usuario, son llevados a través del consumo restful del webserver por medio de jQuery al puerto serial del servidor para la posterior escritura de la orden en el controlador (Arduino) para la ejecución del comando.

La interface de control del robot humanoide es una página HTML que está integrada por el sistema de control de robot, sistema de generación de flujo de video, sistema de adquisición de datos del robot, modelo 3D y la animación del robot. El control del robot humanoide se realiza a través de la utilización de nueve comandos que permiten realizar el control de todas las acciones que tiene el robot.

Los comandos disponibles para el usuario son: movimiento adelante, atrás, derecha, izquierda, mirar derecha, mirar izquierda, stop, activar modo automático, desactivar modo automático y visualizar. Los comandos de control de la interfaz se los presenta en la figura 22.

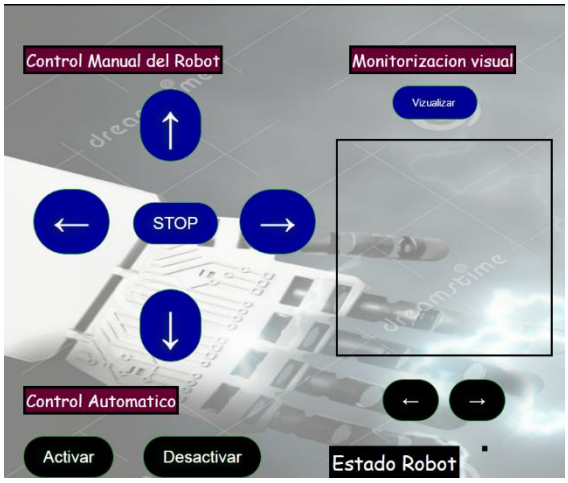


Figura 22. Panel de control del robot humanoide

4.1.2 RASPBERRY PI 2

La placa electrónica utilizada como servidor y controlador es la Raspberry Pi 2, se puede observar en la figura 13. Esta placa fue seleccionada por su multifuncionalidad, ya que básicamente es un minicomputador de escritorio con todas sus características. En el sistema, la Raspberry levanta dos servicios, los cuales están esperando la petición del cliente tanto para la transmisión de video como para la lectura y escritura de los parámetros de control que lee y escribe al arduino nano a través del puerto serial.

4.1.2.1 Sistema de adquisición de datos

La aplicación de control del robot humanoide localizado en la UTE requiere un control en lazo cerrado para determinar la posición real en la que se encuentra el robot, por este motivo se necesita una señal de retroalimentación en el sistema que proporcione la posición del cuerpo del robot respecto a su eje Z para determinar cuándo está caminando o acostado.

Para leer esta señal que proporciona el Arduino nano se utiliza el USB (Universal Serial Bus). Este puerto lo poseen tanto el Arduino nano como el Raspberry Pi,

- **USB**

Universal Serial Bus. Es universal porque puede soportar diferentes tipos de dispositivos, desde los calentadores de café, unidades de disco para adaptadores WiFi, para la reproducción de audio. Este bus serial requiere de cuatro hilos. Fue creado por un grupo de industrias, entre las que se incluyen Compaq, DEC, IBM, Intel, Microsoft, NEC y Northern Telecom. La especificación de USB incluye estandarización de conectores y cables, la topología a través del hub para las conexiones externas con más de 126 dispositivos y los protocolos para el reconocimiento y configuración (ComputerHope, 2016)

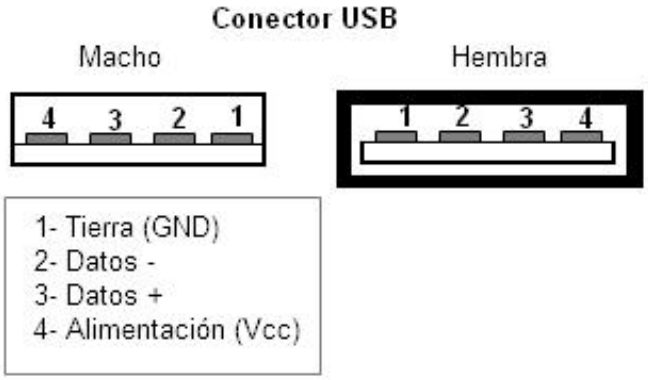


Figura 23. Conector USB
(Electrónica, 2015)

En la figura 23 se puede observar la funcionalidad de los pines del USB, que está compuesto por cuatro pines, donde dos se usan para la alimentación y los dos restantes para la lectura y escritura de los datos independientemente.

Los puertos USB en Raspbian, que es el sistema operativo que utiliza el raspberry llevan otro nombre para estos dispositivos a diferencia de Windows que se los llama COM en el administrador de dispositivos del sistema. En Linux se los configura a través de la ruta /dev/tty para el caso del Arduino nano el nombre del dispositivo es ttyUSB0. Con este nombre se realiza las configuraciones del puerto para la escritura y la lectura de los datos del controlador.

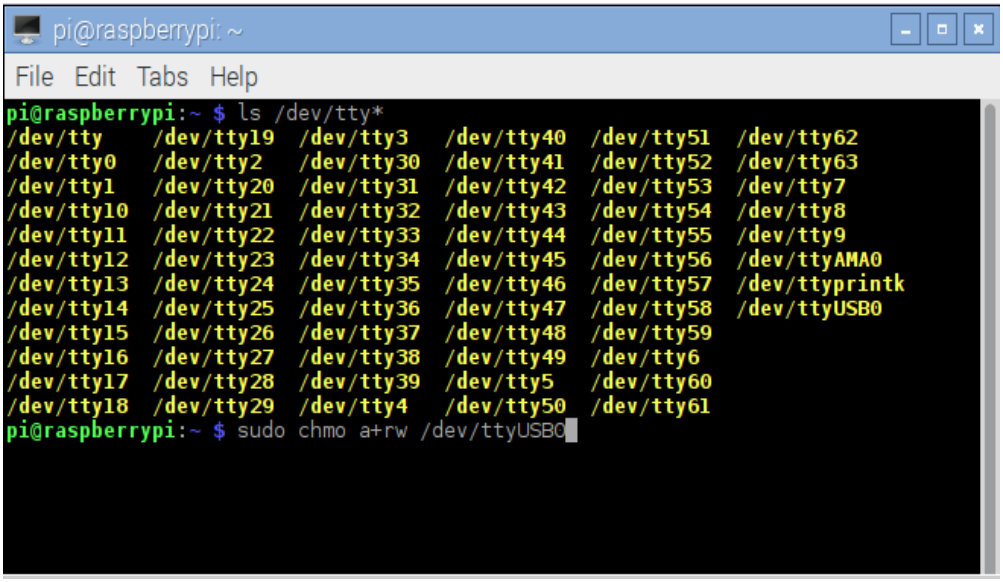


Figura 24. Comandos para identificar permisos de lectura y escritura del puerto

Para la presentación en la interfaz web del ángulo en el cual se encuentra el robot, se desarrolló una función en el back-end la cual abre el puerto serial del servidor, lee lo que está recibiendo y el dato que recibe lo retorna en un formato JSON para poder publicarlo en el serverweb cada vez que sea consumido. La función en el front-end no es consumida por el usuario, está encerrado en bucle el cual consume cada segundo y el dato recibido es presentado en la interfaz.

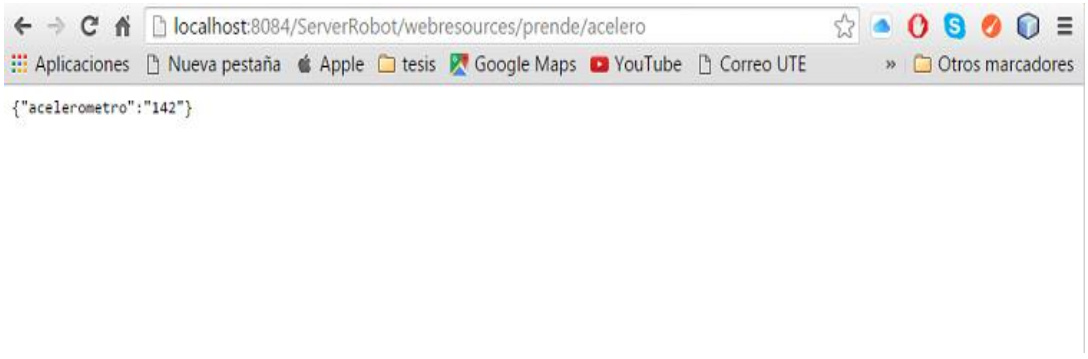


Figura 25. Ángulo del robot formato JSON

Figura 26. Ángulo del robot en la interfaz de control

4.1.2.2 Comunicación Wi-Fi de la placa de control

Para la conexión de la placa de control a la intranet de la UTE se realiza con un dispositivo USB-Wi-fi que conecta directamente al Raspberry. El dispositivo a usar es el USB Edimax EW-7811Un que se lo puede observar en la figura 27.

Figura 27. Conector Edimax EW-7811Un

El dispositivo de la figura 27 es compatible con el Raspberry ya que no se requiere instalar drivers adicionales al sistema operativo para su funcionamiento. Las especificaciones técnicas del Edimax se pueden observar en el anexo 1.

4.1.2.3 Configuración del Servidor para el Webserver

En el Raspebrry Pi pueden ser levantados algunos servidores para tener aplicaciones o páginas web, en este caso se levantará el servidor tomcat7 que viene en los repositorios del raspbian, este servidor es compatible con las librerías y repositorios del lenguaje de programación Java; la aplicación que contiene el webserver está desarrollada en java. La instalación de tomcat7 en el raspbian se realiza a través de la ventana de comandos como se lo presenta en la figura 28.

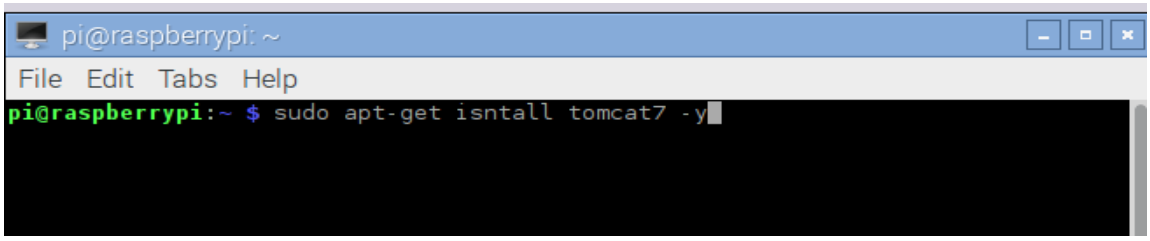


Figura 28. Comando para instalar tomcat en raspbian

Una vez instalado tomcat7 se puede comprobar ingresando al localhost en el browser del Raspberry o de cualquier dispositivo conectado a la red a través de la dirección IP que posee el Raspberry.Para saber la dirección IP del Raspberry se utiliza el comando que se presenta en la figura 29.

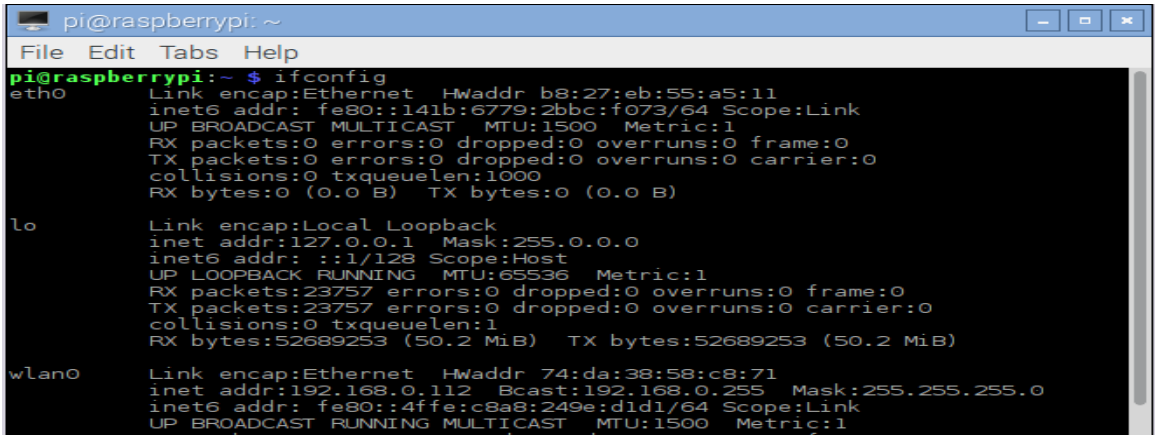


Figura 29. Comando para obtener la dirección IP

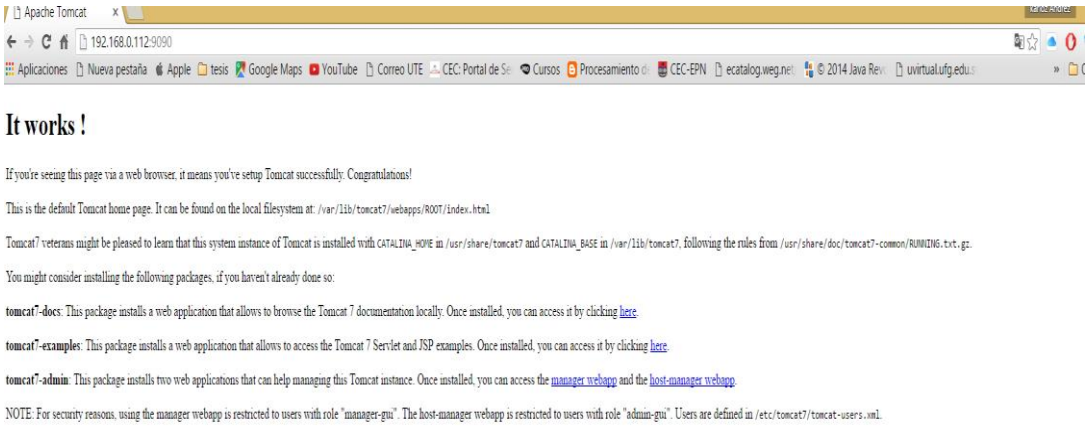


Figura 30. Comprobación del Servidor levantado en la dirección IP del raspberry

El puerto del servidor por el cual se accede a la información también puede ser editado en el archivo de configuraciones de tomcat7. Este puerto se cambia ya que por omisión siempre está configurado en el 8080, pero por la transmisión de la cámara que está instalada en el mismo puerto y para no tener problemas se cambia al 9090.

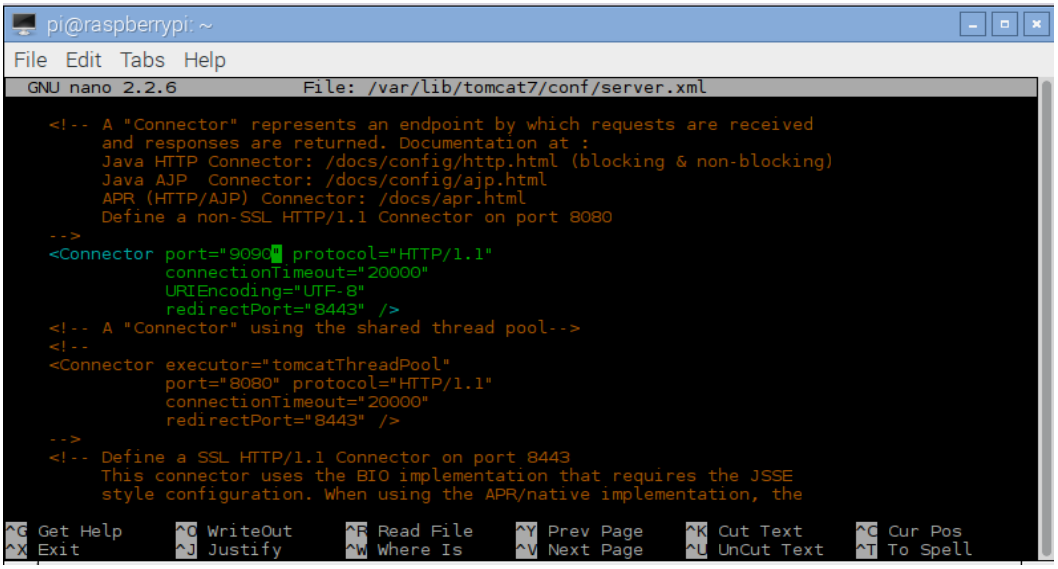


Figura 31. Cambio de Puerto de enlace del servidor

En la figura 31 se presenta el cambio de puerto 8080 a 9090 para el acceso a los archivos del servidor.

Para subir aplicaciones al servidor se debe obtener el archivo con extensión .War de la aplicación java, este archivo se debe guardar en la carpeta WebApps del directorio /var/lib/tomcat7 para poder obtener una ruta de acceso pública a través de la IP del Raspberry, en la figura 32 se observa la ruta por la cual se puede acceder a la aplicación en el servidor web.

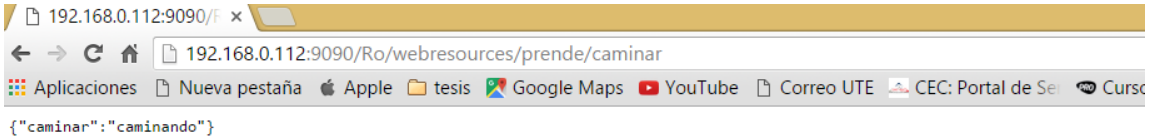


Figura 32. Ruta de acceso al webserver de la aplicación

4.1.2.4 Sistema de Generación de flujo de video a través de Cámara Web

El sistema de generación de flujo de video, permite enviar desde el servidor hacia el operador un flujo de video continuo; lo que observa el robot humanoide, para el desarrollo de este sistema se utilizaron herramientas de código libre como Raspberry y su servicio motion para disminuir costos de desarrollo, el sistema está compuesto por los siguientes elementos:

Un servicio web de flujo de video que lleva el video desde la cámara web hasta el servidor de video para que puede ser utilizada por el navegador del usuario.

Una interfaz web la cual controla la visualización de la señal de video a través de una función de visualización así como también su tamaño y estándar de compresión (Freire, 2013).

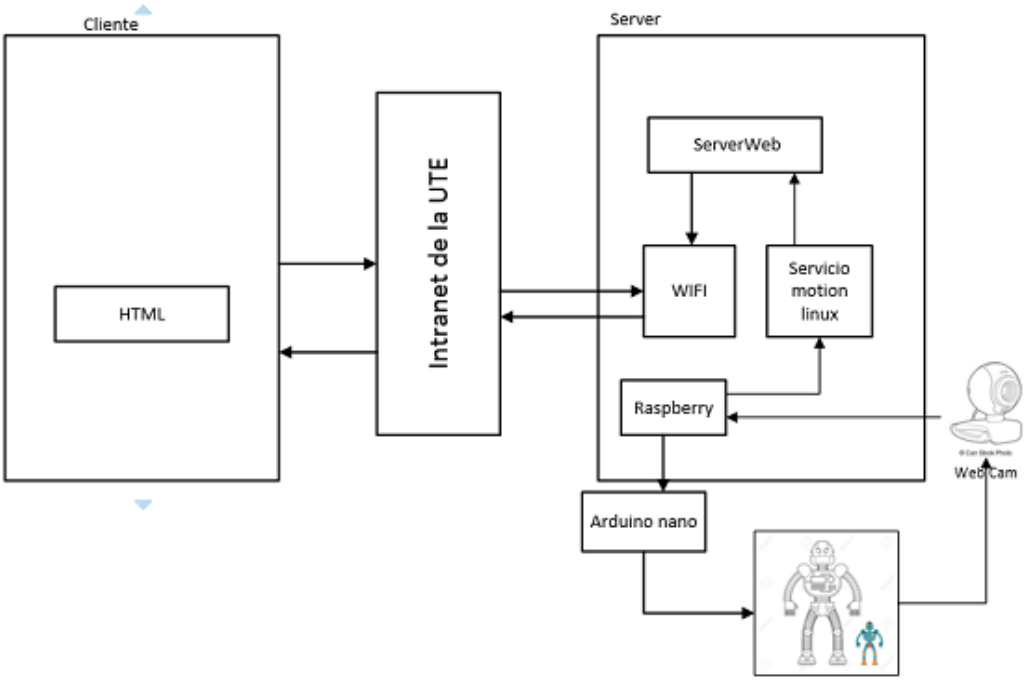


Figura 33. Arquitectura del Sistema de generación de flujo de video

En el Raspberry también se pueden levantar servicios que no contengan una aplicación, así como levantar un servicio de video que transmita las imágenes capturadas por una cámara web conectada a través de USB en el Raspberry. La cámara utilizada para la transmisión de video es la LifeCam HD-300 de Microsoft, esta cámara es compatible con el Raspberry y no necesita instalar drivers adicionales ya que raspbian posee en sus repositorios, en al anexo 2 es posible observar las especificaciones técnicas de la cámara.

Figura 34. Cámara LifeCam HD-300 de Microsoft

Para levantar el servicio de video en la raspberry es necesario instalar el servicio motion de raspbian que permite montar las imágenes captadas por la

cámara en la dirección IP del raspberry. En la figura 35 se presenta la línea de comandos para instalar el servicio motion en raspbian.

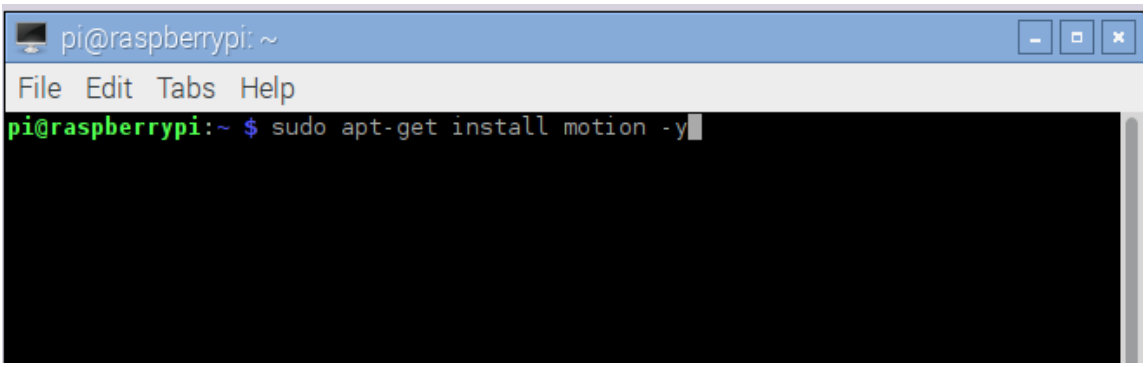


Figura 35. Comando para instalar motion en raspbian

Como el servicio está montado en la dirección IP del Raspberry también se puede modificar el puerto por el cual es accesible, así como las configuraciones de la cámara, resolución, dimensiones, capturas, estas configuraciones se realizan en el archivo montion.conf del directorio /etc/motion como se presenta en la figura 36. También se configura la inicialización automática del demonio de este servidor para que se active al momento de encender el Raspberry en el archivo motion del directorio /etc/default como se presenta en la figura 37.

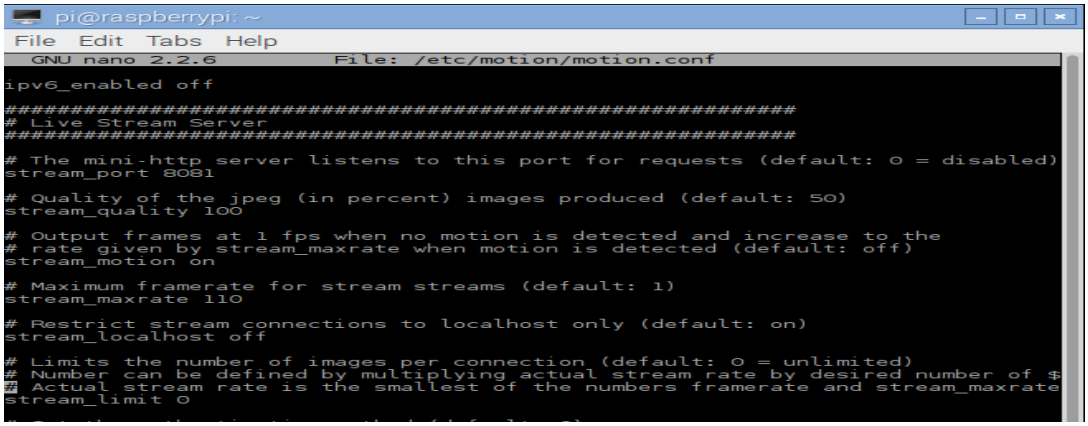


Figura 36. Configuraciones de la visualización de la cámara

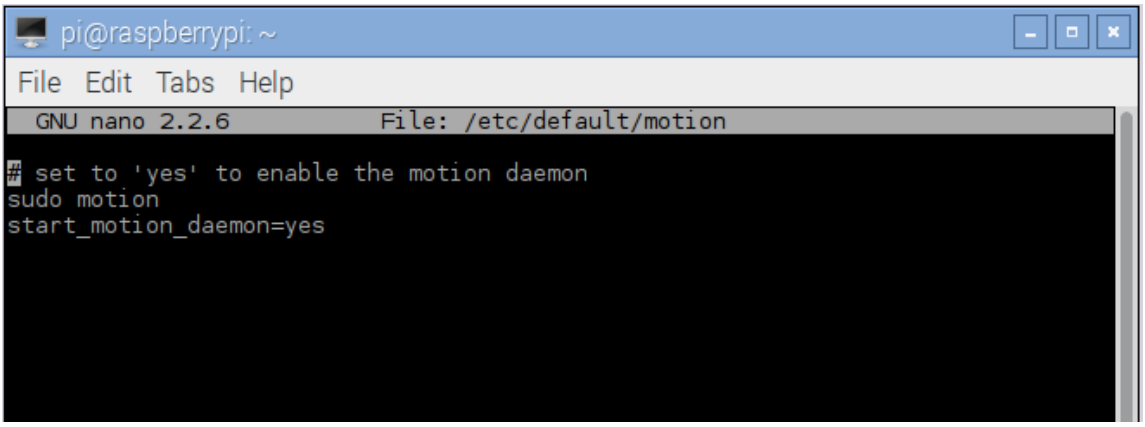


Figura 37. Activación del demonio del servidor

4.2 DISEÑO DEL SOFTWARE

El diseño de software es el proceso de definición de métodos de software, funciones, objetos y la estructura general y la interacción de su código de manera que la funcionalidad resultante pueda satisfacer las necesidades de los usuarios. En general, el diseño debe ser bien pensado y revisado antes de iniciar la codificación. Es más fácil de probar diferentes diseños en el Front-end y descubrir los problemas tempranamente en el ciclo de desarrollo de software que limitarse a un cambio de diseño importante después de que la mayor parte del código se ha escrito.

4.2.1 DISEÑO FRONT-END

El “front-end” es una mezcla de programación y diseño que potencia los efectos visuales y las interacciones de la aplicación entre el usuario y la aplicación web.

4.2.1.1 Html5

Para el desarrollo de la aplicación de control se utilizó HTML5, que es la última versión de HTML. Este lenguaje es el principal para el desarrollo de la estructura de las páginas web, se basa en tags o etiquetas que definen el tipo de elemento en la página web, para el desarollo de una página web con HTML es necesario conocer sus componentes, etiquetas que se muestran en el anexo 3.

La estructura de la aplicación web desarrollada se muestra en la figura 38 y 39.

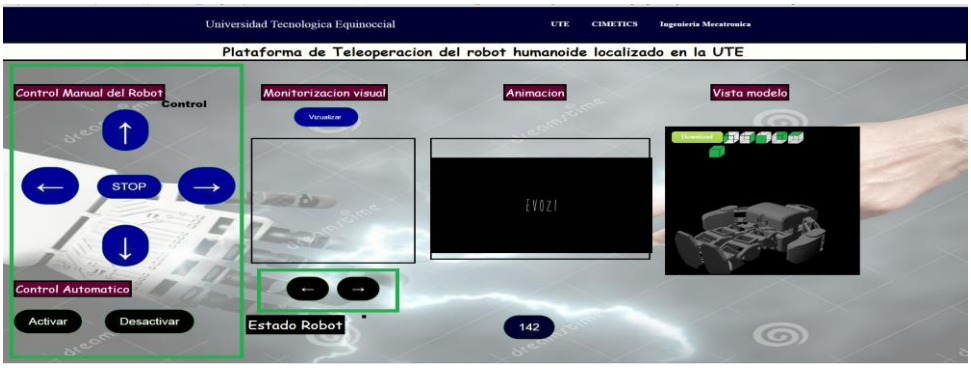


Figura 38. Parte de control de la aplicación web.

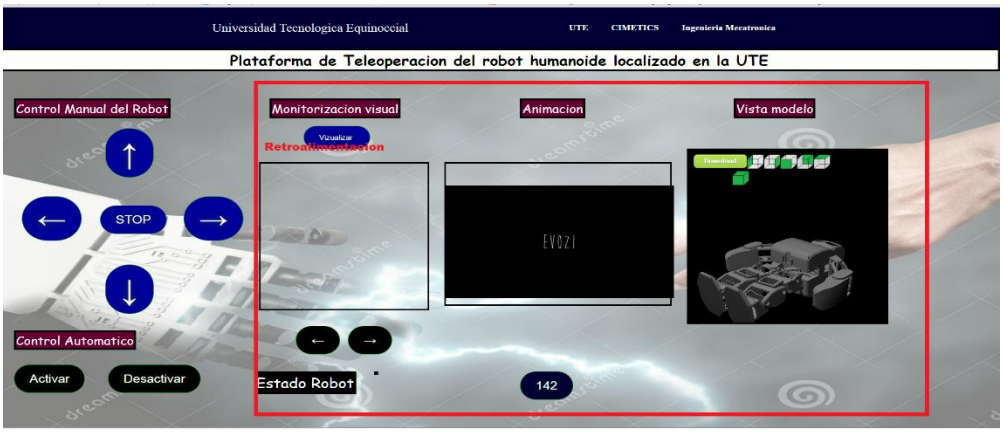


Figura 39. Parte de retroalimentación de la aplicación web

De las imágenes anteriores se puede visualizar la estructura de la aplicación web; mientras que el código completo HTML se puede observar en el anexo 4, que consta de las siguientes partes:

- Encabezado
- Panel de control
- Retroalimentación visual
- Simulación de modelo en 3D
- Representación de datos del sensor

4.2.1.2 CSS

CSS (Hoja de estilo de cascada) es un lenguaje utilizado en el desarrollo del front-end para páginas web; este lenguaje define los estilos de los

componentes de la página como fondos, colores, formas, animaciones, así como el posicionamiento de los elementos de la página web.

Hay varias formas de programar en CSS en una página web, una de ellas es escribiendo un script dentro de la página HTML y referenciando los nombres de los componentes; otra, es direccionando con un enlace a una hoja de estilos con extensión .css que se encuentra en la misma carpeta de la página HTML.

La sintaxis de CSS es:
Nombre para identificar elemento: propiedad
Por ejemplo para definir el color de fondo del elemento body de la página HTML es:

```
Body {
background:#000033;
}
```

El código CSS empleado en la aplicación web de este proyecto desarrolla la presentación de la página así como la animación de los botones, posición de los elementos, fondo de la página y colores de los elementos. Para mantener un mismo estilo con la página principal de la UTE (CIMETICS) se encuentran las siguientes líneas de código:

```
header{
width: 100%;
overflow: hidden;
background:#000033;
margin-bottom: 0,5px;
}
header nav{
        float: right;
        line-height: 90px;
}
```

Este código representa las configuraciones del encabezado de la página, donde se define el color, tamaño, y margen. También está establecido que los elementos que se encuentran dentro del encabezado sean flotantes a la derecha y tenga una distancia del margen superior de 90px, esto quiere decir que siempre se encontrará en la parte derecha del encabezado aunque se reduzca o aumente el tamaño de la página.

Otro estilo importante es el que se da al h1 de la página web:

```
h1{
    text-align: center;
    color: black;
    font-family:cursive ;
    font-size: 30px;
    border: 4px solid #000000;
}
```

El estilo definido en el código anterior representa al título principal de la página web que se configuró como centrado, con borde, con tamaño de letra 30 px y estilo cursiva negrita.

Figura 40. Estilo de header y h1 en la página web

El código CSS completo de la aplicación web se lo muestra en el anexo 5

4.2.1.3 JavaScript

JavaScript es un lenguaje de programación utilizado para hacer páginas web interactivas. Se ejecuta en la computadora del visitante y no requiere descargas constantes de su sitio web. JavaScript se utiliza a menudo para crear script para las páginas web; este lenguaje soporta estilos de programación funcional orientada a objetos.

Para el desarrollo de esta aplicación web, el Script cumple tres funciones principales:

- Programar el comportamiento de los botones que oprima el usuario.

- Embeber el modelo virtual, simulación en la aplicación web.
- Consumir el webServer del servidor.

Para realizar las tres funciones principales las cuales están desarrolladas en JavaScript se usa una sintaxis general donde se define una función la cual según el evento realiza una acción como lo vemos a continuación:

```
function Automatico(){
   $(document).ready(function(){

$.getJSON("http:///10.10.28.240:9090/ServerRobot/webresources/prende/caminar", function(result){
                    $.each(result, function(i, state){
                $("p").html("");
              $("p").append(state + " ");
          });
          });
});
}
```

En el código anterior se presenta la sintaxis de programar una función, donde se antepone la palabra function y el nombre que se le quiere dar y dentro de los corchetes de esta función se coloca la acción a realizar. También se puede observar dentro del código donde se utiliza el método getJSON de la API Jquery que sirve para consumir un servicio tipo rest y obtener los valores de este servicio en el formato JSON para así imprimir posteriormente en la página HTML

Para llamar a la función desde algún elemento del HTML se debe colocar el nombre de la función dentro de su tag respectivo como se presenta en la siguiente línea de código.

```
<a href="#" class="Activar" onclick="Automatico()">Activar</a>
```

El código JavaScript completo de la aplicación web se muestra en el anexo 6.

4.2.1.4 Modelo 3D

El modelado del robot humanoide se hace en SolidWorks, para realizar la animación al robot se cambia de formato para poder animar en el programa

3dmax. La animación del robot depende de las instrucciones ejecutadas por el usuario en la página web que estarán ligadas a funciones pre-cagardas en el JavaScript que reproducirán el video de la animación, por cada función que se ejecute, el video solo se reproducirá si el controlador del robot recibió la señal.

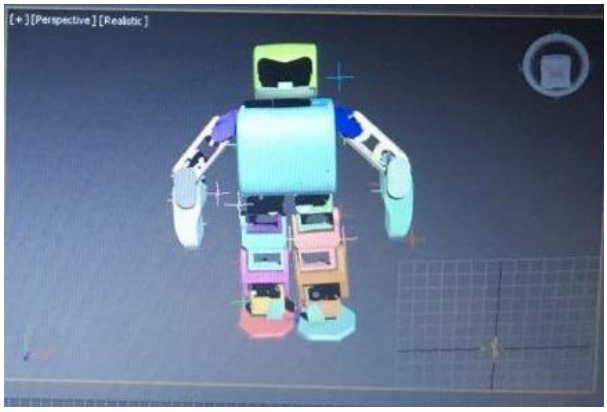


Figura 41. Modelo 3D del robot humanoide .3dmax

Para la presentación del modelo 3D en la interfaz web del robot se utiliza otro programa llamado A3dsViewer el cual permite exportar un modelo 3D a una página HTMlL5, así utilizando el código entrega por el programa e integrarlo al código de la interfaz

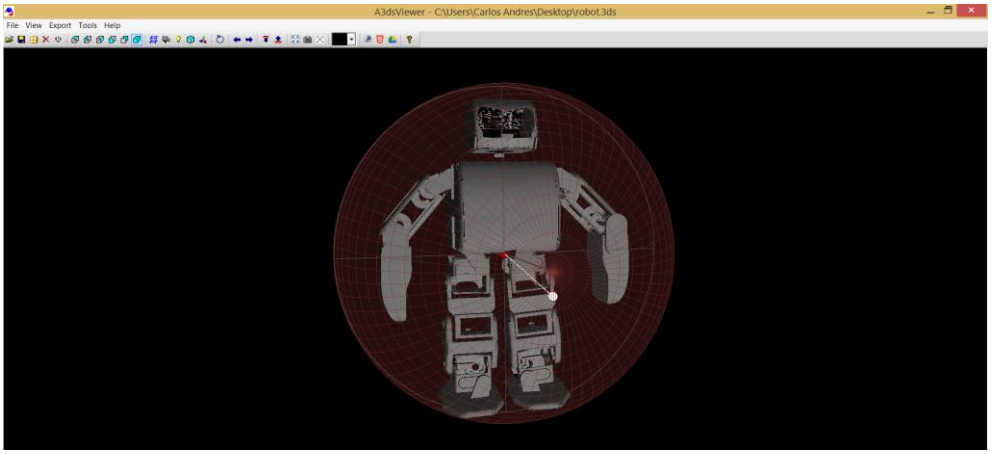


Figura 42. Modelo 3D del robot humanoide .3ds

4.2.2 DIAGRAMAS UML (UNIFIED MODELING LANGUAGE)

Los diagramas UML ayudan a especificar, visualizar y documentar esquemas de sistemas de software, incluyendo su estructura y diseño, de manera que cumpla con todos estos requisitos.

4.2.2.1 Diagrama de Casos de Uso

Modela la funcionalidad del sistema agrupándola en descripciones de acciones ejecutadas por éste para obtener un resultado.

De forma que, muestra el conjunto de casos de uso y actores (un actor puede ser tanto un sistema como una persona) y sus relaciones, es decir, muestra quién puede hacer qué y las relaciones que existen entre acciones (casos de uso). Son muy importantes para modelar y organizar el comportamiento del sistema.

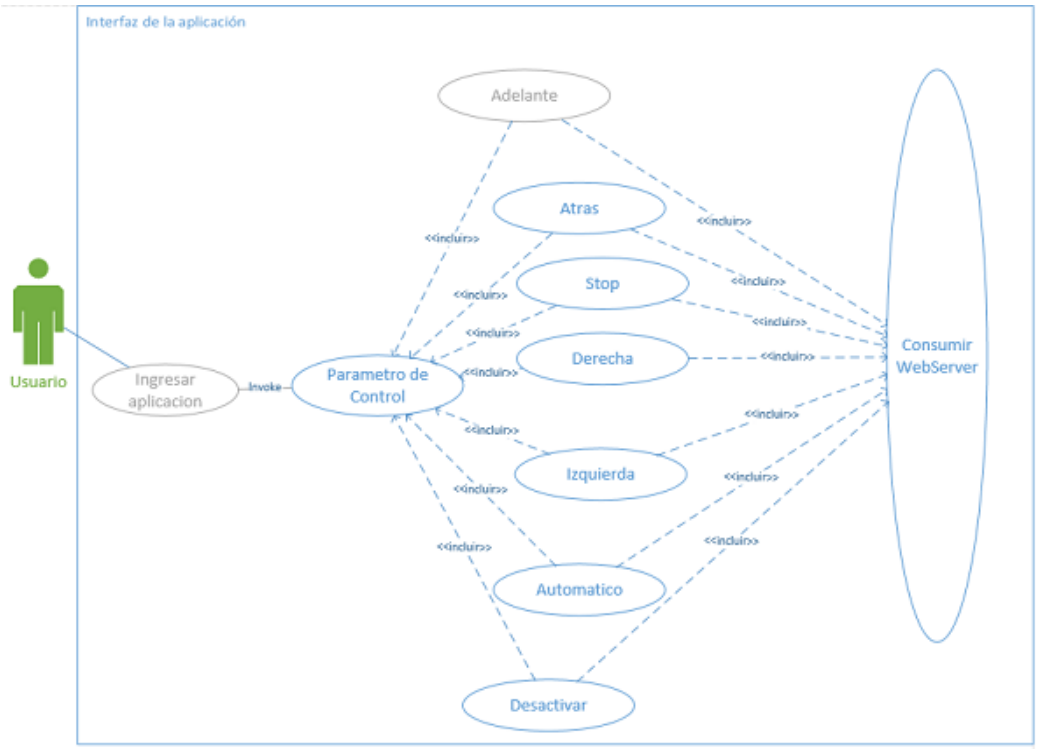


Figura 43. Diagrama caso de usos del Front-end

4.2.2.2 Diagrama de secuencia

El diagrama de secuencia se utiliza principalmente para mostrar las interacciones entre los objetos en el orden secuencial que esas interacciones se producen.

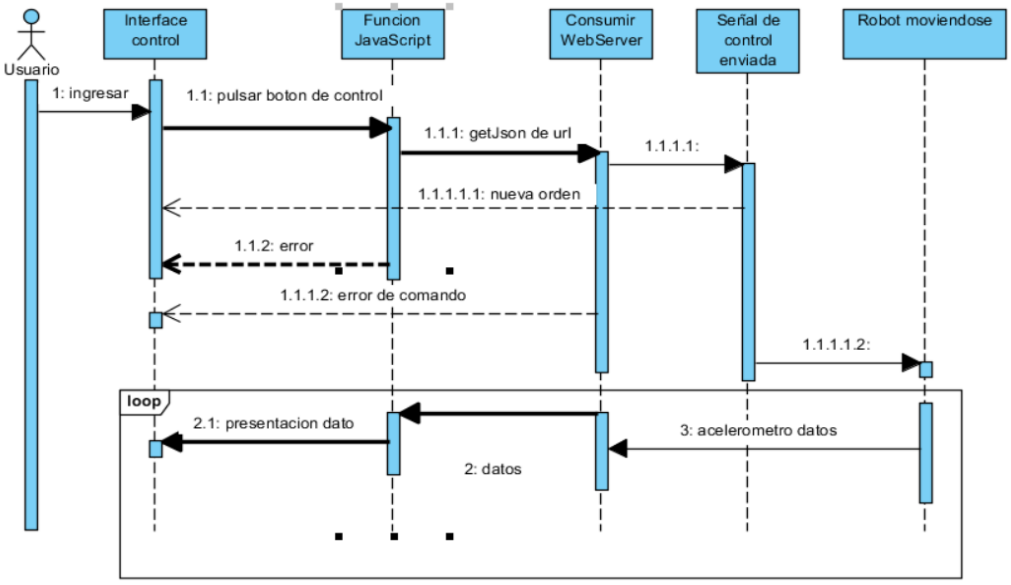


Figura 44. Diagrama de secuencia Front-end

4.2.2.3 Diagrama de estados

El diagrama de estados muestra los diferentes estados que un objeto puede presentar dentro y las transiciones entre estos estados.

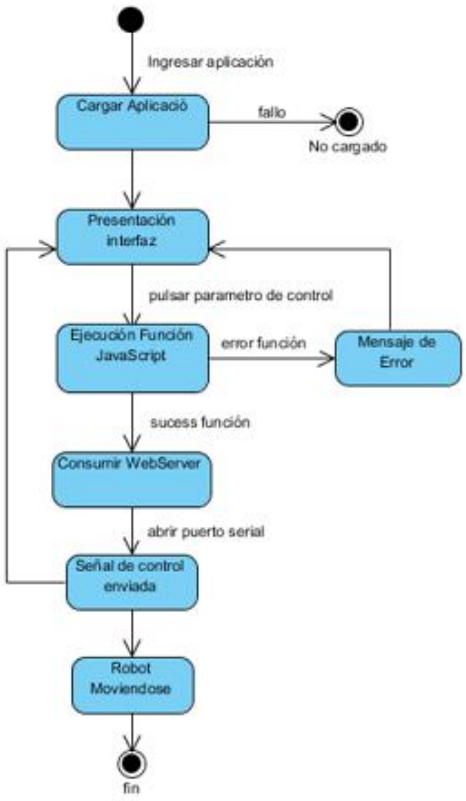


Figura 45. Diagrama de estados Front-end

4.2.3 DISEÑO BACK-END

El "back-end" sirve indirectamente en apoyo de los servicios de front-end, debido a su cercanía con los recursos necesarios, o por su capacidad de comunicarse con los recursos deseados. La aplicación de fondo puede interactuar directamente con el front-end o, quizás más típicamente, es un programa que se llama desde un programa intermedio entre front-end y back-end de actividades.

4.2.3.1 Java

Es el lenguaje de programación orientado a objetos más utilizado en el mundo, es rápido, seguro y fiable. Desde portátiles hasta centros de datos, consolas para juegos, súpercomputadoras, teléfonos móviles e internet, Java está en todas partes (Java, 2016)

Este lenguaje es la herramienta usada para desarrollar los recursos necesarios para la funcionalidad de la aplicación. Al robot se consideró una clase que tendrá sus atributos o funcionalidades que serán consumidas por el usuario. En la figura 46 se presenta la creación del objeto (Robot) y sus atributos (Funcionalidades).

```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package com.funciones;
7
8  import javax.xml.bind.annotation.XmlRootElement;
9
10 /**
11  *
12  * @author Carlos Andres
13  */
14 @XmlRootElement(name="Robot")
15 public class EstadoRobot {
16         private String caminar;
17         private String retroceder;
18         private String parado;
19         private String derecha;
20         private String izquierda;
21         private String camdere;
22         private String camizq;
23         private String malcoman;
24         private String auto;
25         private String acelerometro;
```

Figura 46. Creación de clase Robot

Para la conexión al puerto serial del Arduino nano también se creó una clase para establecer las configuraciones y métodos para la lectura y escritura de las órdenes para el robot.

```
public class SerialClass {

    static SerialPort serialPort;
    public void escribe(String cr){
        SerialPort serialPort = new SerialPort("/dev/ttyUSB0") {}; //
        try {
            System.out.println("Port opened: " + serialPort.openPort());

            try {
                Thread.sleep(700);
            } catch (InterruptedException ex) {
                Logger.getLogger(SerialClass.class.getName()).log(Level.SEVERE, null, ex);
            }

            System.out.println("Params setted: " + serialPort.setParams(9600, 8, 1, 0));
            System.out.println("\"Hello World!!!\" successfully writen to port: " + serialPort.writeBytes(cr.getBytes()));
            System.out.println("\"Hello World!!!\" successfully writen to port: " + serialPort.readString());

            System.out.println("Port closed: " + serialPort.closePort());
        }
        catch (SerialPortException ex){
            System.out.println(ex);
        }
    }


    public String leer(){
        SerialPort serialPort = new SerialPort("/dev/ttyUSB0") {}; ///dev/ttyACM0

        try {
            System.out.println("Port opened: " + serialPort.openPort());

            try {
                Thread.sleep(500);
            } catch (InterruptedException ex) {
                Logger.getLogger(SerialClass.class.getName()).log(Level.SEVERE, null, ex);
            }
```

Figura 47. Creación de clase para el Puerto serial

El código java completo de la clase Robot y SerialClass se muestra en el anexo 7.

4.3 CODIFICACIÓN

La codificación se realiza una vez terminado y realizadas las pruebas con todos los módulos que se diseñaron. Para unificar en un solo modulo los códigos revisados anteriormente desarrollados en java, se embebe al ServerWeb donde estarán instanciados los módulos del robot y del puerto serial como objetos para utilizar sus métodos.

4.3.1 WEB SERVER RESTFUL

Los Servicios tipo Rest son sistemas que utilizan directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquier formato (XML, JSON, etc) sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes.

```
@Path("prende/{param}")
public class WebserverRobot {

    @Context
    private UriInfo context;

    /**
     * Creates a new instance of PrendeResource
     */
    public WebserverRobot() {
    }

    /**
     * Retrieves representation of an instance of com.restful.PrendeResource
     * @param param
     * @return an instance of java.lang.String
     */
      @GET
    @Produces(MediaType.APPLICATION_JSON)
    public EstadoRobot getJson(@PathParam("param") String param) {
        EstadoRobot r = new EstadoRobot();
        SerialClass ser =  new SerialClass();



        //TODO return proper representation object
        if (param.equals("caminar")){
            r.setCaminar("caminando");
            ser.escribe("C");
             return r;

        }else if(param.equals("retroceder")){
            r.setRetroceder("retrocediendo");
            ser.escribe("A");
            return r;
```

Figura 48. WebServer con instancias de Clase Robot y Serial

En la figura 48 se presenta la configuración del ServerWeb donde se establece que recibe un parámetro después del path llamado prende (/prende/"parámetro"). Esta será la ruta con la cual debe consumir el ServerWeb donde dependiendo del parámetro que reciba, realiza la acción de escribir la letra correspondiente en el puerto serial así como retornar la variable en formato JSON para determinar el estado del robot en la página web.

4.3.2 DIAGRAMA DE CLASES

El diagrama de clases describe cómo está conformada la estructura del sistema, sus atributos, operaciones (métodos) y las relaciones que pueden tener los objetos

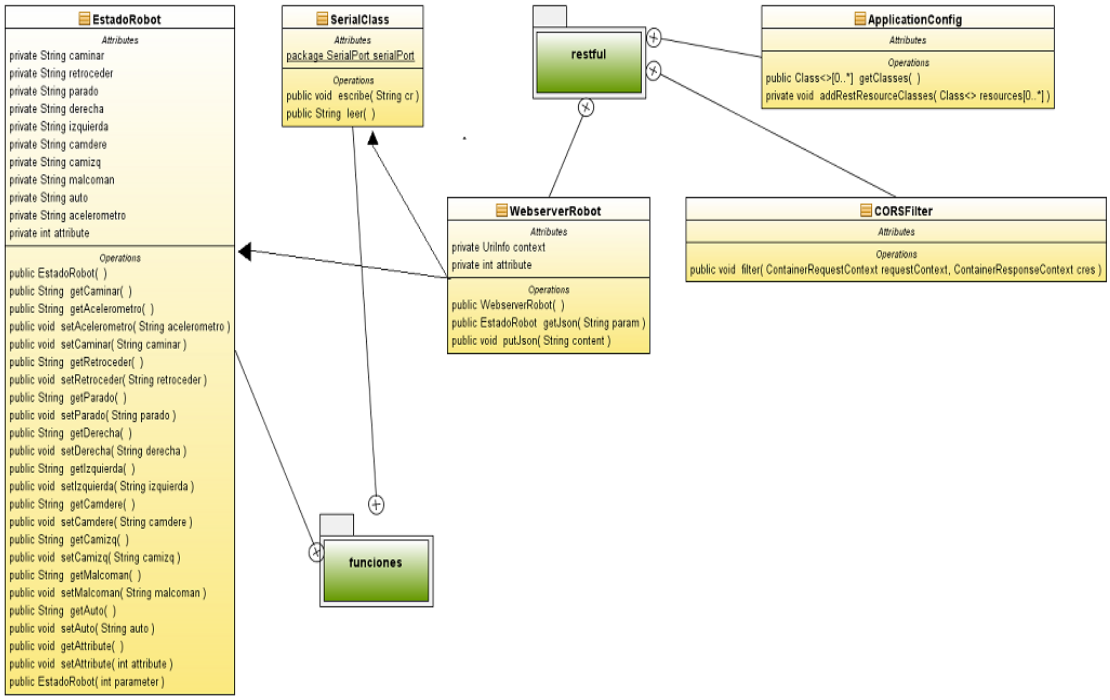


Figura 49. Diagrama de Clases del Back-end

En la figura 49 se observa el diagrama de las clases que componen el back-end de la aplicación web, donde se tiene tres clases importantes que son: EstadoRobot, SerialClass, y WebserverRobot. La clase WebserverRobot tiene una la relación de composición con la clase EstadoRobot y SerialClass ya que es estrictamente necesario que estas clases estén instanciadas en el Webserver para poder utilizar sus métodos y atributos cuando se consuma el server. Las clases ApplicationConfig y CORSFilter están las configuraciones del webserver para su accesibilidad y tipo de respuesta.

5 ANÁLISIS DE RESULTADOS

Para comprobar el funcionamiento de todos los sistemas que integran la plataforma de teleoperación del robot humanoide, se realizaron una serie de pruebas para evaluar los requerimientos planteados inicialmente.

Inicialmente se realizaron pruebas de verificación de la disponibilidad de la aplicación desde los distintos navegadores, también se hicieron pruebas para determinar el tiempo de respuesta del controlador del robot a la orden dada por el operador desde diferentes ubicaciones dentro de la UTE. Por último se evaluó el tiempo de retraso de la trasmisión de la cámara desde diferentes ubicaciones de monitorización y el rendimiento de la CPU del Raspberry Pi.

5.1 PRUEBAS DE DISPONIBILIDAD DE LA APLICACIÓN

Una vez implementados todos los componentes que corresponden a la aplicación se realizan las pruebas de funcionamiento pertinentes para determinar si se logran los objetivos iniciales.

En la primera prueba se realizó la verificación de la disponibilidad de la aplicación web montada sobre el servidor del centro de investigación de mecatrónica (CIMETICS), a donde se accedió desde distintos navegadores de internet como se muestra en las figuras 50, 51,52.

Figura 50. Acceso a la aplicación desde Internet Explorer

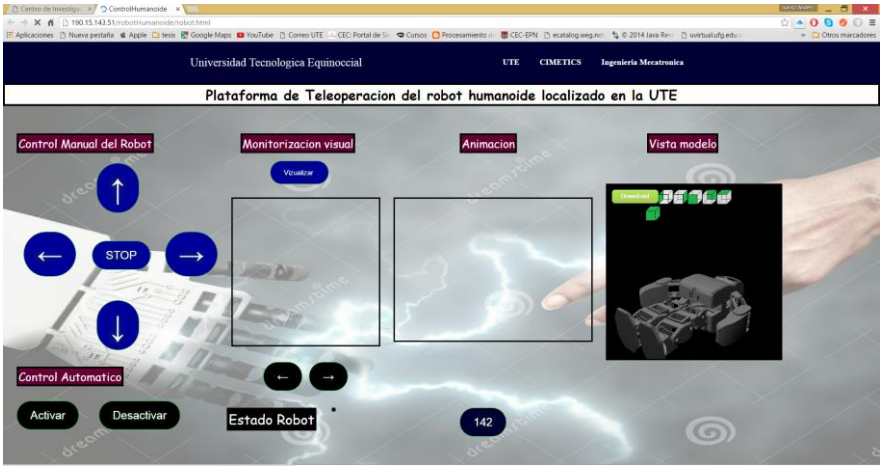


Figura 51. Acceso a la aplicación desde Chrome

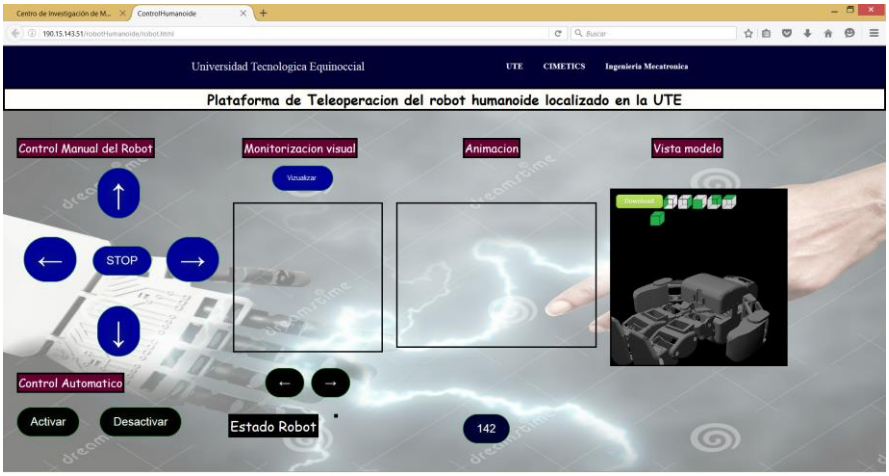


Figura 52. Acceso a la aplicación desde Mozilla Firefox

5.2 PRUEBAS DE TIEMPO DE RESPUESTA DEL CONTROLADOR

Para comprobar la fiabilidad de las órdenes de control y del sistema de comunicación de la aplicación con el robot humanoide, se realizaron pruebas con diferentes ubicaciones del operador. Se evaluó desde diferentes bloques de la Universidad, áreas verdes del recinto, etc. En la tabla 5.1 se pueden observar los resultados obtenidos de las pruebas de control realizadas al robot humanoide.P

Tabla 6. Resultados del tiempo del respuesta del robot

Ubicación	Tiempo de respuesta del Robot
CIMETICS	0.7 segundos
Laboratorios Mecatrónica	0.9 segundo
ITIC	1 segundos
Bar de la Universidad	1 segundos
Áreas verdes de la Universidad	1.2 segundos

Se probaron los botones de control de aplicación comenzando por el modo manual y posteriormente por el modo automático. Cada uno de estos modos de operación de control tiene que enviar la letra correcta para activar una acción en el controlador del robot humanoide, esta prueba se realizó prendiendo y apagando un led a modo de ilustración como se observa en la figura 53.

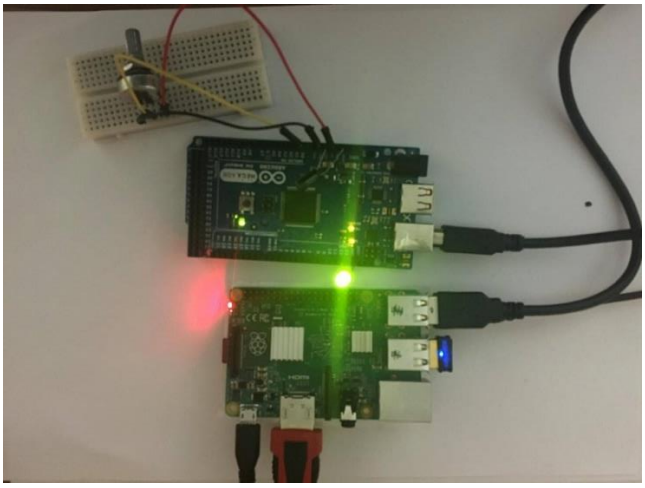


Figura 53. Prueba de activación de placa de control

5.3 PRUEBAS DEL RETRASO DEL VIDEO

Para comprobar la fiabilidad de la trasmisión de video y del sistema de comunicación con el robot humanoide, se realizaron pruebas con diferentes ubicaciones geográficas del observador. Se evaluó desde diferentes bloques

de la Universidad, áreas verdes de la universidad, etc. En la tabla 7 se pueden observar los resultados obtenidos de las pruebas de trasmisión de video.

Tabla 7. Resultados de prueba de trasmisión de video

Ubicación	Tiempo de retraso
CIMETICS	1 segundo
Laboratorios Mecatrónica	1.2 segundos
ITIC	1.5 segundos
Bar de la Universidad	1.3 segundos
Áreas verdes de la Universidad	1.9 segundos

Tabla 8. Características de la trasmisión de video

Formato	MJPEG
Resolución	320*360 pixeles
Fps	60
Stream_port	8081

En la tabla 5.3 se observa las características con las que se trasmite el video en la interfaz web. El tiempo en establecer la comunicación de la trasmisión de video depende del tiempo de respuesta del servidor web donde está levantado el servicio, en la figura 54 se ilustra la transmisión de video dentro la página web.

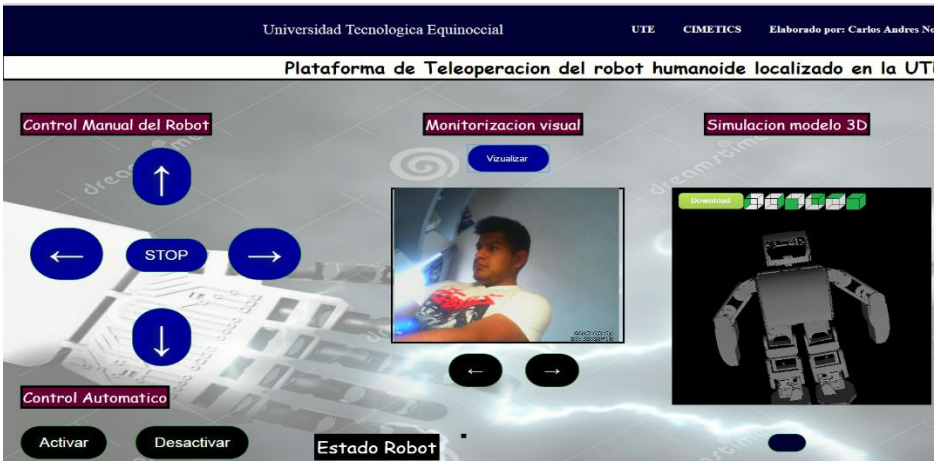


Figura 54. Transmisión de video en la aplicación

5.4 PRUEBAS DE RENDIMIENTO DE LA RASPBERRY PI

Las pruebas para monitorear el rendimiento del Raspberry Pi, se realizaron desde que se inicia el servidor web y todos los servicios en segundo plano que se ejecutan al encender el Raspberry Pi. Inicialmente cuando los servicios son levantados y no se los consume el uso del CPU fue del 3% como se observa en la figura 55.

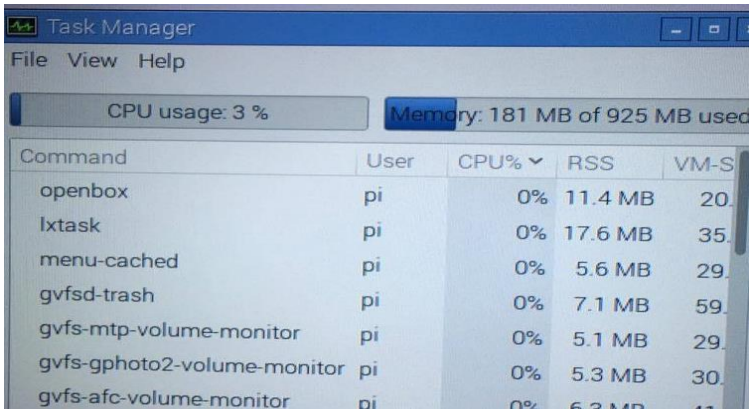


Figura 55 Uso de la CPU al 3%

Cuando el usuario pulsa un comando para mover el robot humanoide, el uso de la CPU subió al 34% como se observa en la figura 56 y luego bajó nuevamente al 3% cuando el usuario dejo de enviar señales de control.

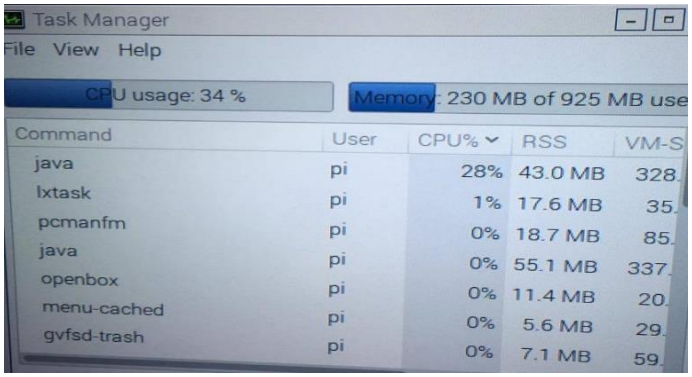


Figura 56 Uso de la CPU al 34%

Cuando el usuario envía las señales de control, activa la cámara, activa la animación del robot el uso del CPU subió al 92% en los cuales los procesos de la aplicación de java son los que aumentan el uso de la CPU como se observa en la figura 57.

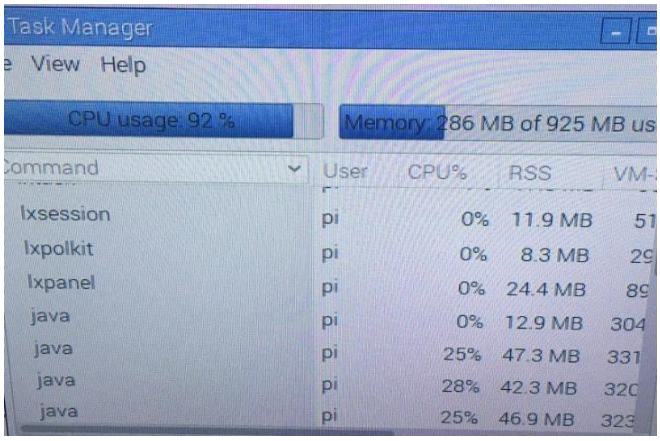


Figura 57. Uso de la CPU al 92%

5.5 ANÁLISIS DE RESULTADOS

Después de haber realizado las pruebas de cada componente que integra la aplicación de control web bajo distintos escenarios de control, y de evaluar los resultados obtenidos de las pruebas de la aplicación web, se puede concluir que el objetivo del trabajo de titulación se ha alcanzado, debido al logro de la implementación del sistema de control inalámbrico del robot humanoide.

En las pruebas realizadas de compatibilidad de la aplicación con los navegadores se observó que tanto en Mozilla Firefox, Chrome e internet Explorer la aplicación funciona sin ningún problema. Asimismo, se evidenció que en el navegador en cual se demoró menos en cargar la aplicación por completo, fue en Chrome; mientras que en el que se demoró más el cargar fue en Internet Explorer.

La comunicación en tiempo real mediante la intranet de la Universidad Tecnológica Equinoccial entre el usuario de la aplicación web y el servidor web montado en la Raspberry Pi es un parámetro importante para que el usuario pueda controlar y supervisar el funcionamiento del robot humanoide. Tras haber realizado las pruebas de tiempo de respuesta del controlador del robot humanoide ante una orden enviada por el usuario se observó que la primera orden se demora entre 3 a 4 segundos; esta orden establece la comunicación con el controlador del robot, las órdenes de control tanto manual

como automática enviadas después de haber establecido la conexión tienen un tiempo de respuesta menor a 1.5 segundos dependiendo de la ubicación del usuario.

Otro resultado relevante del desarrollo de este trabajo de titulación, es el sistema de visualización a través de la cámara web presenta un retraso que puede variar entre 1 a 2 segundos, dependiendo del porcentaje de conectividad del Raspberry Pi a la red de la universidad, así como de la ubicación del usuario.

En las pruebas de rendimiento del Raspberry Pi se demostró que sus características y componentes físicos son suficientemente capaces de realizar las funciones asignadas sin dificultad (servidor de la aplicación, servidor de video). Se observó que la trasmisión de video y la reproducción de la animación usan el mayor porcentaje de la CPU.

También hay que puntualizar que en el desarrollo de los sistemas tanto de hardware y software que componen este trabajo de titulación se utilizó herramientas libres como Raspberry Pi, java, JavaScript, 3DCloud, etc.

6 CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

- Tras haber desarrollado una aplicación web como medio de control y supervisión de un robot humanoide en la cual se integraron varios lenguajes de programación (HTML5, Javascript, Jquery, CSS, Java), los resultados obtenidos fueron muy favorables en cuanto a la accesibilidad que ofrece para el usuario. Es por esto que las aplicaciones realizadas en dos capas como cliente y servidor, poseen la ventaja en la cual el usuario puede acceder a la aplicación usando cualquier hardware como una laptop, Tablet, Smartphone o cualquier dispositivo electrónico que posea un sistema operativo con un navegador de internet.

- Se determinó que los parámetros mínimos de control necesarios que la interfaz de la aplicación web debe tener integrados son botones de control automático, manual (que incluye los movimientos adelante, atrás, derecha, izquierda y stop), de la visualización de la cámara y su posición, y la presentación grafica del modelo 3D del robot humanoide y su animación.

- Se demostró que es posible desarrollar sistemas funcionales de comunicación tanto para la adquisición y envío de datos de control para sistemas robóticos, mediante el uso de herramientas libres, como es el caso de Raspberry Pi que con su multifuncionalidad como servidor de aplicaciones y video permitió establecer las comunicaciones necesarias entre el robot humanoide y la aplicación web sin tener que instalar programas o archivos especiales en el hardware.

- Se demostró que el sistema de la aplicación web permite un rápido acceso a la interfaz de la aplicación para el envío de las acciones de control.

- El sistema embebido entre el Raspberry Pi y el Arduino nano (placa de control del robot humanoide). Permite que el control del robot funcione solo

cuando el usuario esté conectado a la aplicación web, y el Raspberry Pi esté conectado a la red. El Arduino nano no requiere de una fuente de alimentación ya que está conectado al Raspberry Pi por medio del puerto serial donde también obtiene la energía necesaria para funcionar; por lo que el sistema embebido requiere menos recursos energéticos. Igualmente requiere menos recursos físicos como memoria ya que una parte de la aplicación está alojada en el cliente.

- Se demostró que el hardware que integra al Raspberry Pi es capaz de realizar las funciones designadas, sin usar el 100% de su capacidad de procesamiento ya que el máximo uso del CPU de la Raspberry es de 96% cuando está sometido a todos los requerimientos de control de la aplicación.

RECOMENDACIONES

- La aplicación se puede mejorar, incluyendo al front-end a una metodología responsive design, la cual permitirá una visualización personalizada a cada tipo de dispositivo desde el cual se ingresa a la aplicación web.

- Utilizar un dispositivo wifi de mayor alcance para tener un mayor rango de control del robot humanoide y evitar la pérdida de señales de control por estar muy alejado de la red de la Universidad.

- Realizar una base de datos para guardar la información de control del robot humanoide, en la que se puede realizar un análisis de su distancia recorrida para así tener un historial de las distancias y órdenes enviadas al robot.

- Implementar un formato de mayor compresión del video para bajar el consumo de la memoria de la placa de control y así aminorar el tiempo de retraso del video en la interfaz de la aplicación web.

- Implementar una tarjeta de control que tenga mejores características que favorezcan al rendimiento, como la Raspberry Pi 3 que es la última versión de la Raspberry y posee mejoras muy importantes en cuanto a rendimiento y capacidad procesamiento, con lo cual se disminuye el tiempo de respuesta del servidor y tiempo de retraso de la trasmisión de video.

7 BIBLIOGRAFÍA

BIBLIOGRAFIA

- AirForce. (2010). Airforce Technology. Obtenido de http://www.airforce-technology.com/
- Alencastre M., M. L. (2003). Teleoperating Robots in Multiuser Virtual Environments. Proceedings of the Fourth Mexican International Conference on Computer Science IEEE.
- Aracil, R. (2002). Teleoperacion. III Jornadas de Trabajo Enseñanza vía Internet / Web de la Ingeniería de Sistemas y Automática, Alicante.
- Barnes, A. (1994). Toward Integrated Operator Interface for Advanced Teleoperation under Time- Delay. IEEE/RSJ/GI International Conference on Intelligent Robots.
- Bennet, P. (2002). Robotic Mobile Manipulation Experiments at the U.S Army Maneuver Support Center. Usa: Sandia national laboratories.
- Burnett., C. (2007). Obtenido de https://es.wikipedia.org/wiki/System_on_a_chip#/media/File:ARMSoCBlockDiagram.svg
- Butner S. E., G. M. (10 de 2003). Transforming a Surgical Robot for. IEEE Transactions on Robotics and.
- Ceron. (2005). Sistemas roboticas teleoperados. Colombia: Universidad Militar Nueva Granada.
- ComputerHope. (2016). Free computer help and information. Obtenido de http://www.computerhope.com/jargon/u/usb.htm
- Dialaiti, N. (2002). Teleoperation of a Mobile Robot through Haptic Feedback. IEEE International Workshop 2002 HAVE Haptic Virtual Environments and Their Applications,.
- Escobar, L. A. (2012). ESTUDIO DEL SEGUIMIENTO VISUAL DE ANATOMÍAS ORIENTADO A LA ROBOTICA. Obtenido de Repositorio universidad politecnica de Catalunya: http://repositorio.educacionsuperior.gob.ec/bitstream/28000/351/1/T-SENESCYT-0121.pdf

- Fong, T. (2001). Vehicle Teleoperation Interfaces . Autonomous Robots 11 Kluwer Academic Publishers.
- Freire, F. (2013). Tele-Control de Robots Moóviles desde internet. EIDOS, 44-47.
- Futuba. (2015). Futuba. Obtenido de http://www.futaba.com/
- G. Niemeyer, S. J. (1998). Towards Force-Reflecting Teleoperation Over the Internet. International Conference on Robotics and Automation.
- Galiana, I. (2012). Arquitectura de Control Bilateral para Manipulación Remota en Instalaciones de Fusión Nuclea. Obtenido de National Instruments: http://sine.ni.com/cs/app/doc/p/id/cs-14427#prettyPhoto
- Geovanny Argudo, Adrian Arpi, Eduardo Calle. (10 de 2013). Teleoperación de un robot móvil para ambienteshostiles mediante un sistema de control basado ensoftware libre. Obtenido de www.researchgate.net: https://www.researchgate.net/publication/257938389_Teleoperacion_de_un_robot_movil_para_ambientes_hostiles_mediante_un_sistema_de_control_basado_en_software_libre
- Ghodoussi M., B. S. (11-05 de 05 de 2002). Robotic Surgery – the transatalntic Case. IEEE International Conference on Robotics and Automation. ICRA 02.
- Golberg, K., & M., M. (1995). Desktop Teleoperation via the World Wide Web. EEE Intl. Conf. on Robotics and Automation.
- IDF. (2016). Instituto de Diseño y Fabricacion. Obtenido de http://www.institutoidf.com/index.php/es/investigacion/capacidades-del-idf
- Ifremer, F. (2013). IFremer. Obtenido de http://www.ifremer.fr/flotte/systemes_sm/engins/victor.htm#systeme
- Java. (2016). Java. Obtenido de https://www.java.com/es/download/faq/whatis_java.xml
- Lang, S. N. (2003). IN:SHOP Using Telepresence and Immersive VR for a New Shopping Experience. Obtenido de http://bluec.ethz.ch/publications/inshop.pdf.

- Merlin. (2004). Merlin informatik. Obtenido de http://www.merlin.informatik.uni-wuerzburg.de/
- Michael Erazo, D. M. (2009). Diseño e Implementacion de un robot movil Teleoperado en base al Reconocimiento de forma y movimiento de Objetos. Obtenido de https://www.dspace.espol.edu.ec/bitstream/123456789/1062/1/1863.pdf
- NASA. (2016). National Agency of Space Administrator. Obtenido de Jet Propultion Lab: www.nasa.gov
- Navarro, R. (2006). Modelado y Diseño e implementacion de servicios Web. Obtenido de http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf
- Ollero, B. (2001). Robótica; manipuladores y robots móviles. Alfaomega-marcombo.
- Ortega, R. (2009). Servidores Web. Ecuador.
- Padilla, N. (2015). Modelos de ciclo de vida del software. Obtenido de http://www.nelsonpadilla.net/2015/03/17/modelos-ciclo-vida-desarrollo-software/
- Raspberry. (2016). Raspberry. Obtenido de https://www.raspberrypi.org/products/raspberry-pi-2-model-b/
- Robotnik. (2016). Robotnik. Obtenido de http://www.robotnik.es/robots-moviles/guardian/
- Robotoy. (2002). Robotoy. Obtenido de http://robotoy.elec.uow.edu.au/roboframe.html
- Rodriguez, E. M. (2011). Teleoperacion de un robot mindstorm mediante tecnicas de vision artificial. Obtenido de http://e-archivo.uc3m.es/bitstream/handle/10016/12663/PFC_Eduardo_Mendez_Rodriguez_Presentacion.pdf?sequence=3
- Sandia. (2014). Periodico Tech. Obtenido de http://www.sandia.gov/media/periodic/STech/ST2000v2.pdf
- Sauer P., K. K. (2002). Telerobotic simulator in minimal invasive surgery. Proceedings of the Third International Workshop on Robot Motion and Control.

- Sayers C. P., P. R. (1998). Teleprograming for Subsea Teleoperation Using Acustic. IEEE Journal of Oceanic Engineering. Obtenido de IEEE Journal of Oceanic Engineering.
- Superrobotica. (2016). SuperRobotica. Obtenido de http://www.superrobotica.com/S300167.htm
- TechBlog. (2016). Tech Blog. Obtenido de http://www.techeblog.com/index.php/tech-gadget/feature-engineer-builds-real-life-humvee-transformer-robot
- Telegarden. (1995). Telegarden . Obtenido de http://telegarden.aec.at/cgi-bin/gard-control/G?
- WikimediaCommons. (2016). Wikimedia commons. Obtenido de https://commons.wikimedia.org/wiki/Category:Robots
- WikiRobotica. (2016). Wiki de Robotica. Obtenido de http://wiki.robotica.webs.upv.es/wiki-de-robotica/introduccion/clasificacion-de-robots/
- Zeus. (2004). ZEUS. Obtenido de http://www.computermotion.com,

ANEXOS

ANEXO 1

Especificaciones Técnicas Edimax EW-7811Un

EDIMAX
NETWORKING PEOPLE TOGETHER

www.edimax.com

N150 Wi-Fi Nano USB Adapter
Ideal for Raspberry Pi

EW-7811Un

FEATURES AND TECHNICAL SPECIFICATIONS

- Complies with wireless 802.11b/g/n standards with data rate up to 150Mbps
- Green Power Saving : supports smart transmit power control and auto-idle state adjustment
- Increases wireless coverage 3 times* further
- Includes multi-language easy setup wizard
- Supports 64/128-bit WEP, WPA , WPA2 encryption and WPS-compatible
- Supports QoS-WMM, WMM-Power Save mode

TECHNICAL SPECIFICATIONS

HARDWARE INTERFACE	STANDARD	FREQUENCY BAND
1 USB 1.0/2.0 Type A Internal Antenna	IEEE802.11b, 802.11g, 802.11n	2.4000~2.4835GHz (Industrial Scientific Medical Band)
DATA RATE	**INSTALLATION**	**LED & DIMENSION**
11b: 1/2/5.5/11Mbps 11g: 6/9/12/24/36/48/54Mbps 11n (20MHz): MCS0-7 (up to 72Mbps) 11n (40MHz): MCS0-7 (up to 150Mbps)	Multi-language easy setup wizard with auto run configuration	Link/Activity 7.1(H) x 14.9 (W) x 18.5 (D) mm
SECURITY	**OUTPUT POWER**	**HUMIDITY & TEMPERATURE**
WEP 64/128, WPA and WPA2 Software WPS configuration	11b:17 ±1.5dbm 11g:b:15 ±1.5dbm 11n:14 ±1.5dbm	Operating : 10~90% (Non Condensing) Storage : Max. 95% (Non Condensing) Operating : 32~104°F (0~40°C) Storage : -4~140°F (-20~60°C)
SYSTEM REQUIREMENTS	**RECEIVE SENSITIVITY**	**CERTIFICATIONS**
Windows XP/Vista/7/8/8.1/10 Linux & Mac OS	11n(20MHz)@MCS7: -68dBm±2dBm 11n(40MHz)@MCS7: -64dBm±2dBm 11g@54Mbps: -71dBm±2dBm 11b@11Mbps: -81dBm±2dBm	CE, FCC , WiFi

APPLICATION DIAGRAM

An example of how the EW-7811Un can be setup:

- Connects the EW-7811Un wireless adapter to your computer.
- Setups the wireless connection by running the multi-language easy setup wizard.
- Connects to 802.11b/g/n wireless access point or broadband router.

Ideal for Raspberry Pi/Pi 2
Plug & Play!

RoHS FC CE

www.edimax.com

ANEXO 2

Especificaciones técnicas de LifeCam HD-300 de Microsoft

Microsoft®
LifeCam
HD-3000

Version Information	
Product Name	Microsoft® LifeCam HD-3000
Product Version	Microsoft LifeCam HD-3000
Webcam Version	Microsoft LifeCam HD-3000
Product Dimensions	
Webcam Length	1.55 inches (39.3 millimeters)
Webcam Width	1.75 inches (44.5 millimeters)
Webcam Depth/Height	4.28 inches (109 millimeters)
Webcam Weight	3.17 ounces (89.9 grams)
Webcam Cable Length	59.1 inches (1500 millimeters)
Compatibility and Localization	
Interface	High-speed USB compatible with the USB 2.0 specification
Operating Systems[1]	• Microsoft Windows® 8.1, Windows 8, Windows RT 8.1, Windows RT 8, and Windows 7 • Macintosh OS X v10.7-10.9 • Android 3.2 and 4.2 [1]Advanced functionality not available with all devices and/or operating systems. See compatibility information at: www.microsoft.com/hardware/compatibility.
Top-line System Requirements	Requires a PC that meets the requirements for and has installed one of these operating systems: • Microsoft Windows 8.1, Windows 8, or Windows 7 **For VGA video calling:** • Intel Dual Core 1.6 GHz or higher • 1 GB of RAM • 1.5 GB • USB 2.0 required • Windows-compatible speakers or headphones **For 720p HD recording:** • Intel Dual Core 3.0 GHz or higher • 2 GB of RAM • 1.5 GB • USB 2.0 required • Windows-compatible speakers or headphones You must accept License Terms for software download. Please download the latest available software version for your OS/Hardware combination Internet access may be required for certain features. Local and/or long-distance telephone toll charges may apply. Software download required for full functionality of all features. Internet functions (post to Windows Live™ Spaces, send in e-mail, video calls), also require: Internet Explorer® 6/7/8 browser software required for installation; 25 MB hard drive space typically required (users can maintain other default Web browsers after installation)
Compatibility Logos	• Compatible with Microsoft Windows 8 and Windows RT • Optimized for Microsoft Lync • Skype Certified
Software Localization	Microsoft LifeCam software version 3.0 may be installed in Simplified Chinese, Traditional Chinese, English, French, German, Italian, Japanese, Korean, Brazilian Portuguese, Iberian Portuguese, Russian, or Spanish. If available, standard setup will install the software in the default OS language. Otherwise, the English language version will be installed.
Windows Live™ Integration Features	
Video Conversation Feature	Windows Live call button delivers one touch access to video conversation
Call Button Life	10,000 actuations
Webcam Controls & Effects	LifeCam Dashboard provides access to animated video effects and webcam controls
Windows Live Integration Features	Windows Live Photo Gallery integration - Take a photo with LifeCam Software, then with one click open Photo Gallery to edit, tag and share it online Windows Live Movie Maker integration - Record a video with LifeCam Software and start a movie project on Movie Maker with just one click to then upload it to your favorite networking site
Imaging Features	
Sensor	CMOS sensor technology
Resolution	• Motion Video: 1280 X 720 pixel resolution* • Still Image: 1280 X 800
Imaging Rate	Up to 30 frames per second
Field of View	68.5° diagonal field of view
Imaging Features	• Digital pan, digital tilt, vertical tilt, swivel pan, and 4x digital zoom • Fixed focus from 0.3m to 1.5m • True Color - Automatic image adjustment with manual override • 16:9 widescreen • 24-bit color depth
Product Feature Performance	
Audio Features	Integrated microphone
Microphone Technology	Omni directional microphone
Frequency Range	Frequency range 200Hz – 20kHz
Mounting Features	Flexible universal attachment base
Storage Temperature & Humidity	-40 °F (-40 °C) to 140 °F (60 °C) at <5% to 85% relative humidity (non-condensing)
Operating Temperature & Humidity	32° F (0° C) to 104 °F (40 °C) at <5% to 80% relative humidity (non-condensing)
Certification Information	
Country of Manufacture	People's Republic of China
ISO 9001 Qualified Manufacturer	Yes
ISO 14001 Qualified Manufacturer	Yes
Restriction on Hazardous Substances	This device complies with all applicable worldwide regulations and restrictions including, but not limited to: EU directive 2002/95/EC on the Restriction of the Use of Certain Hazardous Substances in Electrical and Electronic Equipment and EU Registration Evaluation and Authorization of Chemicals (REACH) regulation regarding Substances of Very High Concern.
FCC ID	This device complies with part 15 of the FCC Rules and Industry Canada ICES-003. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation. Tested to comply with FCC standards. For home and office use. Model number: 1492, LifeCam HD-3000.
Agency and Regulatory Marks	• ACMA Declaration of Conformity (Australia and New Zealand) • ICES-003 report on file (Canada) • EIP Pollution Control Mark, EPUP (China) • CE Declaration of Conformity (European Union) • WEEE (European Union) • VCCI Certificate (Japan) • KCC Certificate (Korea) • GOST Certificate (Russia) • CITC Letter (Kingdom of Saudi Arabia) • UkrSEPRO Certificate (Ukraine) • FCC Declaration of Conformity (USA) • UL and cUL Listed Accessory (USA and Canada) • CB Scheme Certificate (International)
Windows Certification Kit (WCK)	ID: 1608508 (32-bit) and 1608509 (64-bit) Microsoft Windows 8.1

ANEXO 3

Lista de etiquetas de HTML5

Elemento raíz		
Etiqueta	Descripción	Atributos principales
html	engloba todo el documento	lang

Metadatos		
Etiqueta	Descripción	Atributos principales
head	delimita el encabezado del documento	
title	título del documento (se muestra en la pestaña del navegador)	
base /	URI base para direcciones relativas	href, target
link /	enlace a otros archivos (hoja de estilo, etc.)	href, rel, media, type, title
meta /	metainformación sobre el documento	name, content, charset
style	hoja de estilo incluida en el documento	type, type, title

Secciones		
Etiqueta	Descripción	Atributos principales
body	delimita el cuerpo del documento	
article	artículo	
section	sección	
nav	navegación	
aside	lateral	
h1 a h6	encabezado (de nivel 1 a 6)	
header	cabecera	
footer	pie	
address	dirección (información sobre el autor)	

Contenido (bloque)		
Etiqueta	Descripción	Atributos principales
p	párrafo	
hr /	separador	
pre	texto preformateado	
blockquote	cita larga (que incluye varios párrafos)	cite
ol	lista ordenada	reversed, start, type
ul	lista no ordenada	

li	elemento de lista (ordenada o no ordenada)	value
dl	lista de definición	
dt	término en lista de definición	
dd	definición en lista de definición	
figure	ilustración	
figcaption	pie de ilustración	
div	división	
main	principal	

Texto (en línea)

Etiqueta	Descripción	Atributos principales
a	hiperenlace	href, target, download, rel, type
em	énfasis	
strong	importante	
small	comentario	
s	incorrecto	
cite	obra	
q	cita	cite
dfn	definición	title
abbr	abreviatura	title
data	datos	value
time	fecha y hora	datetime
code	código (de programa de ordenador)	
var	variable (de programa de ordenador)	
samp	salida (de programa de ordenador)	
kbd	teclado	
sub	subíndice	
sup	superíndice	
i	tecnicismo	
b	atención	
u	sonido inarticulado	
mark	resaltado añadido posteriormente, no en el original	
ruby	notación ruby	
rb	elemento de notación ruby	

rt	elemento de notación ruby	
rtc	elemento de notación ruby	
rp	elemento de notación ruby	
bdi	ignorar dirección de escritura	dir
bdo	dirección de escritura	dir
span	otros significados	
br /	salto de línea	
wbr	posible salto de línea	

Modificaciones		
Etiqueta	Descripción	Atributos principales
ins	texto insertado	cite, datetime
del	texto borrado	cite, datetime

Contenido incrustado		
Etiqueta	Descripción	Atributos principales
img /	imagen	alt, src, usemap, ismap, width, height
iframe	marco incrustado en el documento	src, srcdoc, name, sandbox, width, height
embed		src, type, width, height
object	objeto	data, type, width, height
param /	parametro para <objeto>	name, value
video		src, poster, preload, autoplay, loop, muted, controls, width, height
audio		src, preload, autoplay, loop, muted, controls
source		src, type
track		kind, src, srclang, label, default
map	mapa de imagen	name
area /	área en mapa de imagen	alt, coords, href, hreflang, rel, shape, target, type

Tablas		
Etiqueta	Descripción	Atributos principales
table	tabla	border
caption	leyenda de tabla	
colgroup	grupo de columnas	span
col	columna	span
tbody	cuerpo de tabla (grupo de filas)	

thead	cabecera de tabla (grupo de filas)	
tfoot	pie de tabla (grupo de filas)	
tr	fila	
td	celda	colspan, rowspan, headers
th	celda de cabecera	colspan, rowspan, headers, scope, abbr
Formularios		
Etiqueta	Descripción	Atributos principales
form	formulario	accept-charset, action, autocomplete, enctype, method, target
label	etiqueta de un control	form, for
input /	control (hay varios controles: texto, botón radio, etc)	type, name, value, checked, selected, width, height, size, maxlength, ...
button	botón	name, type, value, form
select	menú	name, multiple, size, ...
datalist		
optgroup	grupo de opciones en un menú	label
option	opción de menú	label, selected, value
textarea	área de texto	name, cols, rows, ...
keygen	generador de pares de claves	name, keytype, challenge
output	cálculo	name, for
progress	barra de progreso	value, max
meter	indicador	value, min, max, low, high, optimum
fieldset	grupo de controles	name, disabled
legend	leyenda de grupo de controles	
Scripts		
Etiqueta	Descripción	Atributos principales
script	script	src, type, charset, async, defer
noscript	contenido a mostrar en navegadores que no admiten <script>	
template		
canvas		width, height
Otros		
Etiqueta	Descripción	Atributos principales
!DOCTYPE	tipo de documento (versión de xhtml empleada)	

ANEXO 4

Código completo HTML

```
<!DOCTYPE html>
 <head>
       <title>ControlHumanoide</title>
       <meta charset="UTF-8">
       <meta name="viewport" content="width=device-width, initial-scale=1.0">
       <link rel="stylesheet" type="text/css" href="estilo.css"/>
       <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"></script
>
       <script src="https://d3js.org/d3.v4.min.js"></script>
       <script src="funciones.js"></script>

 </head>
   <header>
    <div class=" wrapper">
       <div class="logo">Universidad Tecnologica Equinoccial</div>
       <nav>
          <a
href="http://www.ute.edu.ec/index.aspx?idSeccion=33&idCategoria=99&idPo
rtal=1">UTE</a>
          <a
href="http://190.15.143.51/CimeticInv/pages/index.xhtml">CIMETICS</a>
          <a
href="https://www.facebook.com/mecatronica.ute?fref=ts">Ingenieria
Mecatronica</a>

       </nav>
     </div>
    </header>

   <body>

   <h1>Plataforma de Teleoperacion del robot humanoide localizado en la
UTE</h1>

   <section id="contenedor" >
       <div class="imagen">
             <img src="fondo.jpg" style="width:1920px;height:785px;">
             <div id="titulo2">Control Manual del Robot</div>
             <div id="titulo3">Control Automatico</div>
             <div id="titulo4">Monitorizacion visual</div>
             <div id="titulo5">Vista modelo</div>
             <div id="titulovideo">Animacion</div>
```

```
<div id="estadoro">Estado Robot</div>

<a href="#" class="myButton" onclick="Adelante()">↑</a>
<a href="#" class="myButton2" onclick="Atras()">↓</a>
<a href="#" class="myButton4" onclick="Derecha()">→</a>
<a href="#" class="myButton5" onclick="Izquierda()">←</a>
<a href="#" class="myButton6" onclick="stop()">STOP</a>
<a href="#" class="Activar" onclick="Automatico()">Activar</a>
<a href="#" class="Desactivar" onclick="stop()">Desactivar</a>
<a href="#" class="camdere" onclick="Camdere()">→</a>
<a href="#" class="camizq" onclick="Camizq()">←</a>
<p id="esta"></p>
<c href="#" class="acel"></c>

<input type="button" class="myButton3" value="Vizualizar " onclick="mostrar()">

<canvas id="myCanvas" width="320" height="320" style="border:3px solid #000000;">
Your browser does not support the HTML5 canvas tag.
</canvas>
<canvas id="myCanvas2" width="380" height="380" style="border:3px solid #000000;">
Your browser does not support the HTML5 canvas tag.
</canvas>
<canvas id="myCanvas3" width="430" height="310" style="border:3px solid #000000;">
Your browser does not support the HTML5 canvas tag.
</canvas>
<video id="video" height="350" width="440">

<source id="mp4" src="Amor.mp4" type="video/mp4" >

<p>Your user agent does not support the HTML5 Video element.</p>
</video>

<div id='oculto' style='display:none;'>
<iframe class="came" src="http://192.168.0.110:8081/?action=stream"frameborder="0" height="320" scrolling="no" width="320">cccc</iframe>
```

```
                </div>
                <div id='oculto2' style='display:block;'>
                    <iframe class="model3D"
                    src='https://googledrive.com/host/0B-
SCnOWjpWxbNHBuVGdkXzgyT0k'
                    frameborder='0'
                    height='380'
                    width='380'
                    ></iframe>
                </div>

        </div>


    </section>
     </body>

</html>
```

ANEXO 5

Código CSS Completo

```
root{
    display: block;

}
*{
margin: 0;
padding: 0;

}
header{
width: 100%;
overflow: hidden;
background:#000033;
margin-bottom: 0,5px;
}

header .logo{
        color:#f2f2f2;
        font-size:25px;
        line-height:90px;
        float:left;
}
header nav{
        float: right;
        line-height: 90px;
}
header nav a{
        display:inline-block;
        color:#fff;
        text-decoration:none;
        padding:10px 20px;
        line-height:normal;
        font-size:18px;
        font-weight:bold;
        -webkit-transition:all 500ms ease;
        -o-transition:all 500ms ease;
        transition:all 500ms ease;

}
header nav a:hover{
        background:#009900;
        border-radius: 40px;

}
```

```
body{

        background: #fffffa;

}
.wrapper{
        width: 90%;
        max-width: 1100px;
        margin:auto;
        overflow:hidden;
}

h1{

        text-align: center;
    color: black;
    font-family:cursive ;
    font-size: 30px;
    border: 4px solid #000000;
}
#contenedor #titulo2{
        position:absolute;
        color:white;
    font-family:cursive ;
    font-size: 25px;
    background: #660033;
    border: 4px solid #000000;
    left:30px;
    top:200px;


}
#contenedor #titulo3{
        position:absolute;
        color:white;
    font-family:cursive ;
    font-size: 25px;
    background: #660033;
    border: 4px solid #000000;
    left:30px;
    top:710px;


}
#contenedor #titulo4{
        position:absolute;
        color:white;
    font-family:cursive ;
```

```
    font-size: 25px;
    background: #660033;
    border: 4px solid #000000;
    left:520px;
    top:200px;


}
#contenedor #titulo5{
        position:absolute;
        color:white;
    font-family:cursive ;
    font-size: 25px;
    background: #660033;
    border: 4px solid #000000;
    left:1410px;
    top:200px;


}
#contenedor #titulovideo{
        position:absolute;
        color:white;
    font-family:cursive ;
    font-size: 25px;
    background: #660033;
    border: 4px solid #000000;
    left:1000px;
    top:200px;


}
#contenedor #estadoro{
        position:absolute;
        color:white;
    font-family:cursive ;
    font-size: 30px;
    background: #000000;
    border: 4px solid #000000;
    left:490px;
    top:800px;


}

#contenedor #esta{
        position:absolute;
        color:black;
```

```
    font-family:cursive ;
    font-size: 25px;
    /*background: #660033;*/
    border: 4px solid #000000;
    left:720px;
    top:800px;


}


.myButton {
	position:absolute;
	top :265px;
	left:205px;
	background-color:#000099;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:65px;
	padding:16px 29px;
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.myButton:hover {
	background-color:#5cbf2a;
}
.myButton:active {
	position:absolute;
	top:267px;
}

.myButton2 {
	position:absolute;
	top : 565px;
	left: 205px;
	background-color:#000099;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
```

```
		color:#ffffff;
		font-family:Arial;
		font-size:65px;
		padding:16px 29px;
		text-decoration:none;
		text-shadow:-2px 1px 24px #2f6627;
}
.myButton2:hover {
		background-color:#bd2c2c;
}
.myButton2:active {
		position:absolute;
		top:568px;

}
.container {
   position: absolute;
}
.myButton3 {
		position:absolute;
		top :260px;
		left:585px;
		background-color:#000099;
		-moz-border-radius:28px;
		-webkit-border-radius:28px;
		border-radius:28px;
		border:1px solid #18ab29;
		display:inline-block;
		cursor:pointer;
		color:#ffffff;
		font-family:Arial;
		font-size:15px;
		padding:16px 29px;
		text-decoration:none;
		text-shadow:-2px 1px 24px #2f6627;
}
.myButton3:hover {
		background-color:#aba537;
}
.myButton3:active {
		position:absolute;
		top:262px;
}

.myButton4 {
		position:absolute;
		top : 415px;
		left: 355px;
```

```
	background-color:#000099;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:55px;
	padding:16px 29px;
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.myButton4:hover {
	background-color:#5cbf2a;
}
.myButton4:active {
	position:absolute;
	top:418px;

}


.myButton5 {
	position:absolute;
	top : 415px;
	left: 45px;
	background-color:#000099;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:55px;
	padding:16px 29px;
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.myButton5:hover {
	background-color:#bd2c2c;
}
.myButton5:active {
	position:absolute;
	top:418px;
```

```
}

.myButton6 {
	position:absolute;
	top : 435px;
	left: 195px;
	background-color:#000099;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:25px;
	padding:16px 29px;
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.myButton6:hover {
	background-color:#422D2D;
}
.myButton6:active {
	position:absolute;
	top:438px;

}
.Activar {
	position:absolute;
	top : 785px;
	left: 30px;
	background-color:#000000;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:25px;
	padding:16px 29px;
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.Activar:hover {
```

```
	background-color:#422D2D;
}
.Activar:active {
	position:absolute;
	top:786px;

}
.Desactivar {
	position:absolute;
	top : 785px;
	left: 210px;
	background-color:#000000;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:25px;
	padding:16px 29px;
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.Desactivar:hover {
	background-color:#422D2D;
}
.Desactivar:active {
	position:absolute;
	top:786px;

}
.camdere {
	position:absolute;
	top : 700px;
	left: 670px;
	background-color:#000000;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:25px;
	padding:16px 29px;
```

```
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.camdere:hover {
	background-color:#422D2D;
}
.camdere:active {
	position:absolute;
	top:699px;

}
.camizq {
	position:absolute;
	top : 700px;
	left: 570px;
	background-color:#000000;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
	color:#ffffff;
	font-family:Arial;
	font-size:25px;
	padding:16px 29px;
	text-decoration:none;
	text-shadow:-2px 1px 24px #2f6627;
}
.camizq:hover {
	background-color:#422D2D;
}
.camizq:active {
	position:absolute;
	top:699px;

}
.acel {
	position:absolute;
	top : 800px;
	left: 1000px;
	background-color:#000033;
	-moz-border-radius:60px;
	-webkit-border-radius:60px;
	border-radius:60px;
	border:1px solid #18ab29;
	display:inline-block;
	cursor:pointer;
```

```
        color:#ffffff;
        font-family:Arial;
        font-size:25px;
        padding:16px 29px;
        text-decoration:none;
        text-shadow:-2px 1px 24px #2f6627;
}


img {
    max-width:100%;
    opacity: 0.5;
}
#myCanvas{

position:absolute;
        top:340px;
        left:500px;
}
#myCanvas2{

position:absolute;
        top:310px;
        left:1320px;
}
#myCanvas3{

position:absolute;
        top:340px;
        left:855px;
}
.came{
        position:absolute;
        top:340px;
        left:505px;
}
.model3D{
position:absolute;
        top:313px;
        left:1323px;

}
video {
        position:absolute;
        top:340px;
        left:855px;

   }
```

ANEXO 6

Código complete JavaScript

```
function Adelante(){

    $(document).ready(function(){



//$.getJSON("http:///10.10.28.240:9090/ServerRobot/webresources/prende/caminar", function(result){

$.getJSON("http://10.10.10.130:9090/ServerRobot/webresources/prende/caminar", function(result){
            $.each(result, function(i, state){
                 $("p").html("");
                $("p").append(state + " ");
                if(state =="caminando"){

                   loadAnotherVideo(1);

                }else{
                    console.log(state);
                }
            });
            })
            .success(function() { console.log("conexion exitosa"); })
            .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })
    });


}
function Atras(){
    $(document).ready(function(){

        $.getJSON("http://
10.10.10.130:9090/ServerRobot/webresources/prende/retroceder",
function(result){
            $.each(result, function(i, state){
                    $("p").html("");
               $("p").append(state+ " ");
               if(state =="retrocediendo"){

                   loadAnotherVideo(2);

                }else{
```

```
                console.log(state);
            }

        });
        })
        .success(function() { console.log("conexion exitosa"); })
        .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })
});

}
function Derecha(){
   $(document).ready(function(){


$.getJSON("10.10.10.130:9090/ServerRobot/webresources/prende/derecha", function(result){
        $.each(result, function(i, state){
                $("p").html("");
            $("p").append(state+ " ");
            if(state =="Camina Derecha"){

               loadAnotherVideo(3);

             }else{
                console.log(state);
            }

        });
        })
        .success(function() { console.log("conexion exitosa"); })
        .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })

});

}
function Izquierda(){
   $(document).ready(function(){

      $.getJSON("http://
10.10.10.130:9090/ServerRobot/webresources/prende/izquierda", function(result){
        $.each(result, function(i, state){
                $("p").html("");
            $("p").append(state+ " ");
            if(state =="Camina Izquierda"){
```

```
                loadAnotherVideo(4);

            }else{
                console.log(state);
            }

        });
        })
        .success(function() { console.log("conexion exitosa"); })
        .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })
});
}
function stop(){
   $(document).ready(function(){

      $.getJSON("http://10.10.10.130:9090/ServerRobot/webresources/prende/stop", function(result){
        $.each(result, function(i, state){
                $("p").html("");
           $("p").append(state+ " ");
           if(state =="Robot Detenido"){

               loadAnotherVideo(5);

            }else{
                console.log(state);
            }

        });
        })
        .success(function() { console.log("conexion exitosa"); })
        .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })
});
}
function Automatico(){
   $(document).ready(function(){

      $.getJSON("http://10.10.10.130:9090/ServerRobot/webresources/prende/automatico",
function(result){
        $.each(result, function(i, state){
                $("p").html("");
           $("p").append(state+ " ");
           if(state =="Estado Automatico"){

               loadAnotherVideo(6);
```

```
            }else{
                console.log(state);
            }


        });
        })
        .success(function() { console.log("conexion exitosa"); })
        .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })
});
}
   function Camdere(){
   $(document).ready(function(){

      $.getJSON("http://
10.10.10.130:9090/ServerRobot/webresources/prende/camdere",
function(result){
        $.each(result, function(i, state){
                $("p").html("");
           $("p").append(state+ " ");

        });
        })
        .success(function() { console.log("conexion exitosa"); })
        .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })

});
}
   function Camizq(){
   $(document).ready(function(){

      $.getJSON("http://
10.10.10.130:9090/ServerRobot/webresources/prende/camizq",
function(result){
        $.each(result, function(i, state){
                $("p").html("");
           $("p").append(state+ " ");

        });
        })
        .success(function() { console.log("conexion exitosa"); })
        .error(function() { alert("El robot se encuentra apagado o el usuario esta fuera de la UTE"); })
});
}
```

```
function acelerometro(){
   $(document).ready(function(){

      //                                                    $.getJSON("http://
10.10.10.130:9090/Ro/webresources/prende/acelero", function(result){
        $.getJSON("http://
10.10.10.130:9090/ServerRobot/webresources/prende/acelero",
function(result){
           $.each(result, function(i, state){
                 $("c").html("");
             $("c").append(state+ " ");

           });
           });
});
}


setInterval(acelerometro, 1300);

function mostrar(){
document.getElementById('oculto').style.display = 'block';}


var videos =
[
 [
   'Video1.mp4'
 ],
 [
   'Video2.mp4'
 ],
 [
   'Video3.mp4'
 ],
 [
   'Video4.mp4'
 ]
 ];


function loadAnotherVideo(n) {
   if (n >= videos.length) n = 0;

   var video = document.getElementsByTagName('video')[0];
   var sources = video.getElementsByTagName('source');
   sources[0].src = videos[n];
```

```
    video.load();
    video.play();

}
```

ANEXO 7

Código back-end

```
package funciones;

import java.util.logging.Level;
import java.util.logging.Logger;
import jssc.SerialPort;
import jssc.SerialPortException;

/**
 *
 * @author Carlos Andres
 */
public class SerialClass {

    static SerialPort serialPort;
    public void escribe(String cr){
        SerialPort serialPort = new SerialPort("COM4") {}; ///dev/ttyUSB0
        try {
            System.out.println("Port opened: " + serialPort.openPort());

            try {
                Thread.sleep(700);
            } catch (InterruptedException ex) {

Logger.getLogger(SerialClass.class.getName()).log(Level.SEVERE, null, ex);
            }

            System.out.println("Params setted: " + serialPort.setParams(9600, 8,
1, 0));
            System.out.println("\"Hello World!!!\" successfully writen to port: " +
serialPort.writeBytes(cr.getBytes()));
            System.out.println("\"Hello World!!!\" successfully writen to port: " +
serialPort.readString());

            System.out.println("Port closed: " + serialPort.closePort());
        }
        catch (SerialPortException ex){
            System.out.println(ex);
        }
    }


    public String leer(){
        SerialPort serialPort = new SerialPort("COM4") {}; ///dev/ttyACM0
```

```
        try {
            System.out.println("Port opened: " + serialPort.openPort());

            try {
                Thread.sleep(500);
            } catch (InterruptedException ex) {

Logger.getLogger(SerialClass.class.getName()).log(Level.SEVERE, null, ex);
            }

            System.out.println("Params setted: " + serialPort.setParams(9600, 8,
1, 0));
           // String dato = serialPort.readString();
            System.out.println("datos de puerto: " +serialPort.readString());
           String dato = serialPort.readString(3);
            System.out.println("datos de puerto: " + dato);


             System.out.println("Port closed: " + serialPort.closePort());
               return dato;

        }
        catch (SerialPortException ex){
            System.out.println(ex);

        }
        return "";
    }



}
```

```
package funciones;

import javax.xml.bind.annotation.XmlRootElement;

/**
 *
 * @author Carlos Andres
 */
@XmlRootElement(name="Robot")
public class EstadoRobot {
         private String caminar;
         private String retroceder;
            private String parado;
         private String derecha;
```

```
    private String izquierda;
    private String camdere;
    private String camizq;
    private String malcoman;
    private String auto;
    private String acelerometro;
     private int attribute;

public EstadoRobot() {
}

public String getCaminar() {
   return caminar;
}

public String getAcelerometro() {
   return acelerometro;
}

public void setAcelerometro(String acelerometro) {
   this.acelerometro = acelerometro;
}

public void setCaminar(String caminar) {
   this.caminar = caminar;
}

public String getRetroceder() {
   return retroceder;
}

public void setRetroceder(String retroceder) {
   this.retroceder = retroceder;
}

public String getParado() {
   return parado;
}

public void setParado(String parado) {
   this.parado = parado;
}

public String getDerecha() {
   return derecha;
}

public void setDerecha(String derecha) {
```

```
    this.derecha = derecha;
}

public String getIzquierda() {
    return izquierda;
}

public void setIzquierda(String izquierda) {
    this.izquierda = izquierda;
}

public String getCamdere() {
    return camdere;
}

public void setCamdere(String camdere) {
    this.camdere = camdere;
}

public String getCamizq() {
    return camizq;
}

public void setCamizq(String camizq) {
    this.camizq = camizq;
}

public String getMalcoman() {
    return malcoman;
}

public void setMalcoman(String malcoman) {
    this.malcoman = malcoman;
}

public String getAuto() {
    return auto;
}

public void setAuto(String auto) {
    this.auto = auto;
}

      public void getAttribute() {
            // TODO - implement EstadoRobot.getAttribute
            throw new UnsupportedOperationException();
      }
```

```
        /**
         *
         * @param attribute
         */
        public void setAttribute(int attribute) {
                this.attribute = attribute;
        }

        /**
         *
         * @param parameter
         */
        public EstadoRobot(int parameter) {
                // TODO - implement EstadoRobot.EstadoRobot
                throw new UnsupportedOperationException();
        }



}

package restful;


import funciones.EstadoRobot;
import funciones.SerialClass;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.UriInfo;
import javax.ws.rs.Produces;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PUT;
import javax.ws.rs.PathParam;
import javax.ws.rs.core.MediaType;

/**
 * REST Web Service
 *
 * @author Carlos Andres
 */
@Path("prende/{param}")
public class WebserverRobot {

    @Context
    private UriInfo context;
        private int attribute;
```

```
/**
 * Creates a new instance of PrendeResource
 */
public WebserverRobot() {
}

/**
 * Retrieves representation of an instance of com.restful.PrendeResource
 * @param param
 * @return an instance of java.lang.String
 */
  @GET
     @Produces(MediaType.APPLICATION_JSON)
public EstadoRobot getJson(@PathParam("param") String param) {
    EstadoRobot r = new EstadoRobot();
    SerialClass ser =  new SerialClass();



    //TODO return proper representation object
    if (param.equals("caminar")){
        r.setCaminar("caminando");
        ser.escribe("C");
         return r;

    }else if(param.equals("retroceder")){
        r.setRetroceder("retrocediendo");
        ser.escribe("A");
        return r;

    }else if  (param.equals("izquierda")){
        r.setIzquierda("Camina Izquierda");
        ser.escribe("I");
        return r;
    }else if  (param.equals("derecha")){
        r.setDerecha("Camina Derecha");
        ser.escribe("D");
        return r;
    }else if  (param.equals("stop")){
        r.setParado("Robot Detenido");
        ser.escribe("S");
        return r;
    }else if  (param.equals("automatico")){
        r.setAuto("Estado Automatico");
        ser.escribe("T");
        return r;
    }else if  (param.equals("camdere")){
        r.setCamdere("mirar derecha");
```

```
            ser.escribe("R");
            return r;
        }else if  (param.equals("camizq")){
            r.setCamizq("mirar izquierda");
            ser.escribe("L");
            return r;
        }
        else if  (param.equals("acelero")){
            r.setAcelerometro(ser.leer());
            return r;
        }
        else {

        r.setMalcoman("Comando no reconocido");

        return r;}
    }

    /**
     * PUT method for updating or creating an instance of PrendeResource
     * @param content representation for the resource
     */
    @PUT
          @Consumes(MediaType.APPLICATION_JSON)
    public void putJson(String content) {
    }
}
```