



RESUMEN

“DISEÑO DE UN SISTEMA DE TELEMEDICIÓN Y TELECONTROL MEDIANTE EL USO DE LOS ESTÁNDARES INALÁMBRICOS GPRS Y BLUETOOTH.”

La tecnología móvil e inalámbrica que nos proveen los modernos teléfonos celulares se asemeja cada vez más a las capacidades de cómputo y comunicación de un computador personal, por tanto cada vez más las capacidades y funcionalidades de los modernos teléfonos celulares son subutilizadas por los usuarios de éstos dispositivos.

El presente sistema telemático, permitirá aprovechar al máximo las capacidades de los móviles para brindar confort y seguridad a un domicilio. Este complejo sistema consta de cinco partes bien diferenciadas: primero un dispositivo móvil celular con soporte para páginas HTML, que es el “cliente lejano”; segundo tenemos un “servidor” que aloja un servlet y una base de datos; tercero, tenemos un dispositivo móvil celular denominado “cliente cercano”, ya que dispone del jsr-82 para una conexión de corto alcance; cuarto, tenemos un móvil celular que hace de “router”, ya que une tres redes inalámbricas, dos conexiones Bluetooth y una GPRS, este componente del sistema es el más complejo, ya que dispone de jsr-82, jsr-120, jsr-135, y conexión GPRS, para comunicación Bluetooth, envío de SMS, captura de imágenes y comunicación con el servlet respectivamente; y como quinto y último componente tenemos el “hardware de control” desarrollado a base de un PIC16F877A conjuntamente con un módulo RN-41 quien le provee comunicación Bluetooth de clase 1.

Este sistema domótico, ha sido simulado con tecnología java tanto para la web como para móviles; como tecnología java para la web tenemos un servlet, que es una aplicación que se ejecuta en el “servidor”, y recibe y responde las peticiones del cliente, y como tecnología para móviles tenemos dos midlets de J2ME, que son los que permitieron simular las funcionalidades del “router” como del “cliente cercano” sobre el sistema operativo Symbian.

Palabras clave: GPRS, Bluetooth, servlet, base de datos, J2ME, Symbian, PIC 16F877A, domótico, SMS, sistema telemático.



SUMMARY

"DESIGN OF A TELEMESURE AND TELECONTROL SYSTEM BY MEANS OF THE USE OF GPRS AND BLUETOOTH WIRELESS STANDARDS."

The mobile and wireless technology that provide us the modern cellphones resemble each time more to the computer and communication capacities of a personal computer, therefore each time the capacities and functionalities of modern cellphones are more underused for devices' owners.

The present telematic system, will allow to take advantage of cellphones maximum capacities to offer comfort and security to a home. This complex system consist of five well differentiated parts: first, a cellular mobile device with support for HTML pages, which is the "distant client"; second, we have a "server" that lodges a servlet and a database; third, we have a mobile cellular device called "near client", since it has the jsr-82 for a short reach connection; fourth, we have a cellphone that works as a "router", since it joins three wireless networks, two Bluetooth and a GPRS connections, this is the most complex component of the system, since it has jsr-82, jsr-120, jsr-135, and GPRS connection for Bluetooth communications, SMS sending capability, still images capturing, and communication with the servlet; finally we have the control hardware developed using a PIC16F877A with a RN-41 module which provides class 1 Bluetooth communication.

This domotic system has been simulated with JAVA technology for the web as well as for mobiles; JAVA technology for the web consist in a servlet, that is an application which is executed in the "server", it receives and responds the client's petitions, and for mobiles technology we have two J2ME midlets that allow to implement the "router's" functionalities as the "near client" above the Symbian operating system.

Key Words: GPRS, Bluetooth, servlet, database, J2ME, Symbian, PIC16F877A, domotic, SMS, telematic system.



Contenido

| | |
|--|----|
| RESUMEN | 1 |
| SUMMARY | 2 |
| CERTIFICACIÓN | 7 |
| AUTORIA | 8 |
| AGRADECIMIENTO | 9 |
| DEDICATORIA | 10 |
| CAPITULO 1. INTRODUCCIÓN | 11 |
| 1.1 ANTECEDENTES..... | 11 |
| 1.2 ESTADO DEL ARTE..... | 11 |
| 1.3 DESCRIPCIÓN DEL PROBLEMA Y/O NECESIDAD | 11 |
| 1.3.1 PROBLEMAS A SER RESUELTOS..... | 11 |
| 1.3.2 NECESIDADES A SER SATISFECHAS. | 12 |
| 1.4 JUSTIFICACIÓN DEL PROYECTO DE TESIS..... | 13 |
| 1.5 OBJETIVOS..... | 13 |
| 1.5.1 OBJETIVOS GENERALES | 13 |
| 1.5.2 OBJETIVOS ESPECÍFICOS | 13 |
| 1.6 ALCANCE DEL PROYECTO..... | 14 |
| 1.6.1 PRESELECCIÓN, ANÁLISIS Y VALORACIÓN DE POSIBLES PROCEDIMIENTOS Y TECNOLOGÍAS A USAR EN LA SOLUCIÓN. | 14 |
| 1.6.2 ELECCIÓN DE LOS PROCEDIMIENTOS Y LAS TECNOLOGÍAS A UTILIZAR..... | 15 |
| 1.6.2.1 <i>Diseño de la solución.</i> | 15 |
| 1.6.3 IMPLEMENTACIÓN DE UN PROTOTIPO..... | 15 |
| MARCO TEÓRICO | 16 |
| CAPITULO 2. DESCRIPCIÓN DE LA TECNOLOGÍA..... | 16 |
| 2.1 DOMÓTICA..... | 16 |
| 2.2 CELULAR DIGITAL..... | 18 |
| 2.3 JAVA ME..... | 19 |
| 2.3.1 CONCEPTO..... | 19 |
| 2.3.2 CLDC..... | 20 |
| 2.3.3 MIDP..... | 21 |
| 2.3.4 JSRS..... | 21 |
| 2.3.5 SERVLETS..... | 22 |
| 2.4 RED GPRS..... | 23 |
| 2.4.1 CONCEPTO..... | 23 |
| 2.4.2 CARACTERÍSTICAS GENERALES DEL GPRS..... | 24 |
| 2.5 TECNOLOGÍA INALÁMBRICA BLUETOOTH..... | 25 |



| | |
|--|-----------|
| 2.6 PIC16F877A..... | 27 |
| 2.7 MICROCODE STUDIO 3.0.0.5 | 28 |
| 2.8 SIMULADOR PROTEUS 7.6..... | 28 |
| 2.9 MÓDULO BLUETOOTH RN-41..... | 28 |
| 2.10 SENSOR DE MOVIMIENTO..... | 29 |
| 2.11 ACTUADORES. | 30 |
| 2.11.1 LUMINARIAS | 30 |
| 2.11.2 CERRADURA ELÉCTRICA..... | 30 |
| 2.11.3 PUERTA AUTOMÁTICA | 31 |
| CAPITULO 3: DISEÑO E IMPLEMENTACIÓN DEL HARDWARE. | 33 |
| 3.1 ANÁLISIS DEL PROBLEMA | 33 |
| 3.2 DIAGRAMA DE BLOQUES DEL HARDWARE DEL SISTEMA | 34 |
| 3.3 DIAGRAMA DE FLUJO DEL FIRMWARE DEL PIC..... | 35 |
| 3.4 CODIFICACIÓN EN MICROCODE STUDIO..... | 36 |
| 3.5 INTERFACE DE COMUNICACIÓN ENTRE EL MÓDULO BLUETOOTH Y EL PIC..... | 37 |
| 3.6 SIMULACIÓN DEL HARDWARE EN PROTEUS 7.6. | 38 |
| 3.7 IMPLEMENTACIÓN. | 40 |
| 3.8 PROBANDO EL HARDWARE. | 43 |
| CAPÍTULO 4: ANÁLISIS DEL PROBLEMA DEL SOFTWARE DEL SISTEMA..... | 47 |
| 4.1 ANÁLISIS DEL PROBLEMA. | 47 |
| CAPÍTULO 5: DISEÑO DEL SOFTWARE DEL SISTEMA..... | 53 |
| 5.1 DISEÑO PARA LA INTERACCIÓN CERCANA. | 53 |
| 5.1.1 CASO DE USO TELECONTROL CERCANO (VÍA BLUETOOTH)..... | 53 |
| 5.1.2 CASO DE USO TELEMEDICIÓN CERCANA (VÍA BLUETOOTH)..... | 54 |
| 5.2 DISEÑO PARA LA INTERACCIÓN LEJANA. | 55 |
| 5.2.1 CASO DE USO TELECONTROL LEJANO (VÍA GPRS). | 55 |
| 5.2.2 CASO DE USO TELEMEDICIÓN LEJANA (VÍA GPRS). | 56 |
| 5.3 DISEÑO PARA REGISTRO DE NUEVOS USUARIOS. | 57 |
| 4.3.1 CASO DE USO REGISTRO LEJANO (VÍA GPRS)..... | 57 |
| 4.3.2 CASO DE USO REGISTRO CERCANO (VÍA BLUETOOTH Y GPRS). | 58 |
| 5.4 SELECCIÓN DEL LENGUAJE Y SGBD. | 59 |
| 5.5 SELECCIÓN DEL DOMINIO Y HOSTING. | 60 |
| 5.6 CONFIGURACIÓN DE HARDWARE Y REDES. | 60 |
| CAPÍTULO 6: IMPLEMENTACIÓN DEL SOFTWARE DEL SISTEMA..... | 61 |
| 6.1 IMPLEMENTACIÓN PARA EL SERVIDOR..... | 61 |



| | |
|---|------------|
| 6.1.1 CODIFICACIÓN DE LA APLICACIÓN WEB EN JAVA. | 61 |
| 6.1.1.1 Lectura de imágenes provenientes del Router. | 61 |
| 6.1.1.2 Lectura y presentación de la imagen guardada en el Servidor. | 62 |
| 6.1.1.3 Tablas de la Base de Datos..... | 64 |
| 6.2 IMPLEMENTACIÓN PARA EL CELULAR ROUTER. | 66 |
| 6.2.1 CODIFICACIÓN DEL MIDLET PARA EL ROUTER..... | 66 |
| 6.2.1.1 Envío de la imagen al servidor desde el Router. | 66 |
| 6.2.1.2 Envío de un SMS de alerta al cliente lejano. | 68 |
| 6.2.1.3 Telemandos de interacción entre el Router y el Servidor. | 71 |
| 6.2.1.4 Telemandos de interacción entre el Router y el Hardware. | 71 |
| 6.3 IMPLEMENTACIÓN PARA EL CELULAR CLIENTE CERCANO. | 86 |
| 6.3.1 CODIFICACIÓN DEL MIDLET PARA EL CELULAR CLIENTE CERCANO. | 86 |
| 6.3.1.1 Telemandos de interacción entre el Cliente cercano y el Router..... | 86 |
| CAPÍTULO 7: PRESUPUESTO..... | 89 |
| 7.1 PRESUPUESTO. | 89 |
| CAPÍTULO 8: CONCLUSIONES Y RECOMENDACIONES..... | 93 |
| 8.1 CONCLUSIONES. | 93 |
| 8.2 RECOMENDACIONES..... | 94 |
| BIBLIOGRAFÍA | 95 |
| INDICE DE FIGURAS | 97 |
| INDICE DE TABLAS | 99 |
| GLOSARIO DE TERMINOS Y ABREVIATURAS | 100 |
| ANEXOS..... | 102 |
| A-1. PLANO DE LA VIVIENDA A AUTOMATIZAR..... | 102 |
| A-2. HOJA DE ESPECIFICACIONES DEL MÓDULO BLUETOOTH RN-41. (ROVINGNETWORKS S.F.)..... | 103 |
| A-3. PINOUT DEL MÓDULO BLUETOOTH RN-41. (ROVINGNETWORKS S.F.) | 104 |
| A-4. HOJA DE ESPECIFICACIONES DEL SENSOR DE MOVIMIENTO. (BRICOGEEK.COM S.F.)..... | 105 |
| A-5. CÓDIGO FUENTE EN PBPV2.44, EDITADO CON MICROCODESTUDIO V3.005..... | 106 |
| A-6. VERIFICACIÓN DE LA DISPONIBILIDAD DEL DOMINIO EN LA WEB..... | 117 |
| A-7. PÁGINA WEB DEL PROVEEDOR DEL SERVICIO DE HOSTING..... | 118 |
| A-8. ACTIVACIÓN DE LA CUENTA PARA EL SERVICIO DE HOSTING. | 119 |
| A-9. SIMULACIÓN DEL SISTEMA TELEMÁTICO APLICADO A LA DOMÓTICA..... | 126 |



UNIVERSIDAD DE CUENCA
FACULTAD DE INGENIERÍA
MAESTRÍA EN TELEMÁTICA



UNIVERSIDAD DE CUENCA
FACULTAD DE INGENIERÍA
MAESTRÍA EN TELEMÁTICA

*“DISEÑO DE UN SISTEMA DE TELEMEDICIÓN Y
TELECONTROL MEDIANTE EL USO DE LOS
ESTÁNDARES INALÁMBRICOS GPRS Y
BLUETOOTH.”*

Tesis previa a la obtención del
grado de Magíster en Telemática.

AUTOR

Ing. Lenin Guaya Delgado.

DIRECTOR

Ing. Juan Andrade Rodas. M.Sc.

Julio, 2011

Cuenca-Ecuador



CERTIFICACIÓN

Ing.

Juan Andrade Rodas.

DIRECTOR DE TESIS

CERTIFICA:

Que el presente trabajo de investigación ha sido minuciosamente revisado, por lo que queda autorizada su presentación.

Cuenca, julio del 2011.

.....

Ing. Juan Andrade Rodas.

DIRECTOR DE TESIS



AUTORIA

LOS CONCEPTOS, INTERPRETACIONES E IDEAS VERTIDAS EN EL PRESENTE INFORME SON PLASMADOS BAJO RESPONSABILIDAD EXCLUSIVA DE SU AUTOR.

.....
Ing. Lenin Guaya Delgado.

EL AUTOR



AGRADECIMIENTO

Dejo constancia de mi más sincero agradecimiento a mis padres Herminio Guaya y Piedad Delgado, excelentes maestros quienes me han enseñado a estudiar y motivarme; así mismo a mis hermanos Mónica, Diego y Richar por su valioso apoyo en esta importante etapa de mi vida.

A la Universidad de Cuenca, que me permitió recibir formación profesional de cuarto nivel para servir a la sociedad.

A la Facultad de Ingeniería de la Universidad de Cuenca que me ha brindado su apoyo para la culminación del presente proyecto.

Al Ingeniero Juan Andrade quien ha sabido dirigir de manera adecuada la presente.

A todas las personas que de una u otra forma han sabido colaborar con sus ideas al desarrollo del presente proyecto.

EL AUTOR



DEDICATORIA

Mi trabajo les dedico a todos quienes considero seres queridos, y a todos mis amigos y amigas leales y sinceros (as).

A todos los idealistas, que ansían como yo, que el desarrollo de las Nuevas Tecnologías de la Información y las Comunicaciones, permitan cerrar la Brecha Digital existente aún en nuestros pueblos...

El Autor



CAPITULO 1. INTRODUCCIÓN

1.1 Antecedentes.

Para el desarrollo del presente sistema de control, se ha creído conveniente aprovechar la red de telefonía celular mediante midlets (aplicaciones para móviles) y ofrecer telemandos a una vivienda; estos telemandos permitirán activar/desactivar los 14 actuadores, que para este ejemplo demostrativo, serán cerraduras eléctricas, reflectores y lámparas fluorescentes; así como también el monitoreo de 4 sensores de movimiento instalados en una vivienda.

De igual manera, aprovechando los módulos Bluetooth presentes en todos los celulares modernos, desde una aplicación de celular y a corta distancia se podrá controlar el hardware y por tanto, todos los dispositivos eléctricos instalados en la vivienda.

1.2 Estado del arte.

1.2.1 Realidad mundial.

Países desarrollados desde hace pocos años atrás ya disfrutan de esta tecnología, ya que su situación económica y fácil acceso a las Nueva Tecnologías de la Información y las Comunicaciones, les permite instalar modernos dispositivos de control y automatización con fines de comodidad y seguridad en sus hogares y negocios.

1.2.2 Situación en el Ecuador.

En nuestro país, es poco común el uso de sistemas de control y automatización en nuestros hogares y negocios, es por ello que se ha creído conveniente desarrollar un sistema de telemedición y telecontrol de bajo coste, que se adapte a las necesidades de nuestro medio, que por lo regular no son tan sofisticadas.

1.3 Descripción del problema y/o necesidad

1.3.1 Problemas a ser resueltos.

Este proyecto se basa en la programación de un sistema micro-procesado independiente del ordenador, que permite la integración de un sistema de monitoreo y control con el



servicio GPRS (General Packet Radio Service) de telefonía celular y con el estándar Bluetooth presente en los terminales móviles modernos.

Haciendo una pequeña introducción a la parte técnica, vale recalcar que este proyecto se caracteriza por el uso de tecnología inalámbrica y móvil para el intercambio de información entre cinco partes del sistema que están bien diferenciadas: tenemos el celular “cliente lejano”, el “servidor”, el “cliente cercano”, el celular “router” y el “hardware de control”, estas cinco partes conforman el sistema de telemedición y telecontrol mediante el uso de los estándares inalámbricos GPRS y Bluetooth.

Este proyecto es una solución práctica para aquellos domicilios en donde sus dueños se ausentan por tiempos sustanciales; y, necesitan simular presencia en la vivienda con el objetivo de persuadir acciones delictivas. Y es así, que el sistema, permite la captura de imágenes en caso de intrusión ya que posee sensores de movimiento, que le indican cuando hacer la captura de imágenes.

Este sistema de control, permite que mediante un comando vía GPRS se abran las puertas de un domicilio; considerando que el costo de los paquetes de transmisión de datos tienden a disminuir, así que esto no representaría un problema económico. Más aún cuando el sistema tiene incorporado tecnología (Bluetooth) de corto alcance, que le permite interactuar con el sistema sin pago alguno.

El sistema restringe vía software, que sólo usuarios autorizados tengan acceso al control de los 18 dispositivos instalados en la vivienda, desde cualquier parte donde exista cobertura celular, y esté disponible el servicio GPRS.

1.3.2 Necesidades a ser satisfechas.

La necesidad del desarrollo de éste tipo de sistemas, radica principalmente en poder brindar el confort y seguridad a una vivienda. Al observar el anexo A-1, nos damos cuenta que al leer el estado de los 4 sensores, y manipular los 14 dispositivos actuadores de la vivienda de una manera sistemática y programada, el sistema brinda confort y seguridad; convirtiendo de ésta manera a nuestro domicilio en una moderna morada.



1.4 Justificación del proyecto de tesis.

El desarrollar este sistema implica conocimientos de, telecomunicaciones, electrónica, informática y computación, ya que es la perfecta unión entre hardware, software y comunicaciones, con la finalidad de dar una solución versátil para una vivienda.

Los beneficios que prestará son los siguientes:

- Permitirá televigilancia del domicilio, ya que el sistema emitirá 4 alarmas diferentes en caso de intrusión, puesto que son 4 los sensores requeridos que se instalarán.
- Permitirá controlar 14 dispositivos eléctricos a distancia mediante el teléfono móvil.
- Permitirá simular presencia.
- Permitirá controlar el hardware desde una aplicación de telefonía móvil.
- Permitirá usar su celular como llave electrónica.
- Permitirá configurar localmente los números autorizados mediante un midlet.
- Permitirá configurar remotamente los números autorizados mediante una página web desde Internet.
- Permitirá la autenticación del usuario para que pueda tener acceso al Hardware de Control.

1.5 Objetivos.

1.5.1 Objetivos generales

- Diseñar un sistema de monitoreo y control para una vivienda, mediante el uso de los estándares inalámbricos GPRS y Bluetooth.

1.5.2 Objetivos específicos

- Diseñar una interface de comunicación entre el módulo Bluetooth y el PIC.



- Controlar la apertura de seis cerraduras eléctricas, con telemandos enviados desde un teléfono celular, mediante el uso de los estándares inalámbricos GPRS y Bluetooth.
- Controlar el encendido y apagado de luminarias incandescentes: R1, R2, R3, como se muestra en el anexo A-1, con telemandos enviados desde un teléfono celular , mediante el uso de los estándares inalámbricos GPRS y Bluetooth.
- Controlar el encendido y apagado de luminarias fluorescentes: L1,L2,L3,L4 y F1, como se muestra en el anexo A-1, con telemandos enviados desde un teléfono celular, mediante el uso de los estándares inalámbricos GPRS y Bluetooth.
- Monitorear el estado de sensores de movimiento: S1, S2, S3, S4, como se muestra en el anexo A-1, mediante el uso de los estándares inalámbricos GPRS y Bluetooth.
- Configurar el hardware localmente mediante un midlet.
- Configurar el hardware remotamente mediante Internet.
- Implementar la autenticación para los usuarios.

1.6 Alcance del proyecto.

Desarrollar un prototipo de hardware de control que permita ser instalado en un domicilio, y permita leer el estado de 4 sensores y controlar 14 dispositivos actuadores desde un celular tanto a corta como larga distancia, mediante tecnología Bluetooth y GPRS respectivamente.

1.6.1 Preselección, análisis y valoración de posibles procedimientos y tecnologías a usar en la solución.

Se podría usar infrarrojo, pero la velocidad de transmisión es muy baja, y está en desuso en los modernos teléfonos celulares.

Se podría usar tecnología Wifi 802.11 a, b, g o n para un sistema de este tipo, pero la distancia que alcanza Bluetooth clase 1, es suficiente para una interacción de corto



alcance con el sistema. Además la tecnología Bluetooth está disponible en gamas baja, media y alta en las diferentes marcas de teléfonos celulares actuales.

Se podría usar tecnología Wimax, pero se incrementaría el costo de instalación de esta infraestructura de red.

Esta solución se basa en el estándar inalámbrico GPRS, porque toda la infraestructura de red ya está instalada y lista para usar; desde luego previa la contratación de un paquete de datos, que hoy a raíz de su difusión en el servicio para smartphones, gradualmente va disminuyendo en precio, lo cual es muy beneficioso para los usuarios de éste sistema que a lo largo de éstos capítulos se detalla.

Esta solución se implementará haciendo uso de equipos basados en el estándar Bluetooth, porque es un estándar mundialmente difundido en los teléfonos móviles celulares actuales de todas las marcas.

Esta solución se basará en un microcontrolador de Microchip, porque existe gran cantidad de información al respecto, además permiten ser codificados en muchos lenguajes de programación, y ser programados por la mayoría de grabadores universales de microcontroladores.

Este sistema permite ser controlado desde el celular y configurado desde un navegador en una PC con Internet; ya que los ordenadores, los celulares y la infraestructura de red que los conecta, son el motivo de la actual revolución de las NTIC.

1.6.2 Elección de los procedimientos y las tecnologías a utilizar.

1.6.2.1 Diseño de la solución.

En la Figura 2-1 que se encuentra a continuación, observamos el diseño de la solución, aquí notamos que la solución está hecha a base de un microcontrolador, el mismo que a través del uso de tecnología GPRS y Bluetooth, envía y recibe datos hacia y desde un teléfono móvil celular tanto a corta como a larga distancia.

1.6.3 Implementación de un prototipo.

El prototipo, es un hardware que a través del uso de tecnología GPRS y Bluetooth permite el envío y recepción de datos. La implementación del prototipo, se la hará a base de un microcontrolador quien por una parte se conecta a los sensores y actuadores, y por otra al Módem Bluetooth, y por intermedio de éste, a la red GPRS de telefonía celular.



MARCO TEÓRICO

CAPITULO 2. DESCRIPCIÓN DE LA TECNOLOGÍA

2.1 Domótica.

En el Diccionario de la Real Academia Española se indica que la palabra domótica proviene del latín domus – casa – y del término informática, siendo el “conjunto de sistemas que automatizan las diferentes instalaciones de la vivienda”. De manera amplia la definición es adecuada, pero en realidad la cuestión va más allá de la mera automatización.

Puede definirse como la opción, integración y aplicación de las nuevas tecnologías informáticas y comunicativas al hogar. Incluye principalmente el uso de electricidad, dispositivos electrónicos, sistemas informáticos, y diferentes dispositivos de telecomunicaciones, incorporando la telefonía móvil e Internet.

Algunas de sus principales características son: interacción, interrelación, facilidad de uso, teleoperación o manejo a distancia, fiabilidad y capacidad de programación y actualización. Su arquitectura puede ser centralizada o distribuida, aunque en realidad, por las ventajas de intercomunicación y ante los fallos, se emplea más la descentralizada.

Los protocolos pueden ser estándar, es decir compatibles entre sí, y propietarios, que son los creados exclusivamente para un cliente o aplicación única. La configuración estándar cuenta con un sistema compuesto por ordenador u ordenadores, módem, tarjeta de sonido, dispositivos de ampliación de audio, baterías de emergencia, sondas de temperatura – exterior e interior, detectores de humo, gas y agua, vídeo portero, sensores magnéticos para puertas y ventanas, detectores de presencia, mandos a distancia y emisores-receptores de señal.

Existen tres tipos de redes domóticas en el hogar según la infraestructura necesaria: las que utilizan nuevos cables, las que emplean los ya existentes – principalmente las redes eléctricas preexistentes – y las que se basan en sistemas inalámbricos o sin cables.

Sus principales prestaciones o funciones son una mayor seguridad, la automatización y el telecontrol de los electrodomésticos y otros dispositivos, el acceso a los nuevos



sistemas de telecomunicaciones y la superior disponibilidad de ocio y entretenimiento en casa. En todos los casos, existe una fuerte tendencia a hacer más cómoda y versátil la estancia en el lugar de la vivienda, al igual que se espera tener una mayor capacidad de gestión y monitoreo, tanto de los electrodomésticos como de los servicios públicos, donde destacan aspectos como el consumo, el gasto y el ahorro energético.

En algunos casos se ha pretendido hacer pasar por sistema domótico algunas aplicaciones a distancia en el hogar, pero la realidad es que hoy se considera como domótico un sistema que integre múltiples servicios y prestaciones, no sólo algunos y por separado. Uno de los principales avances recientes en la materia es la articulación entre los sistemas de telecomunicaciones y los sistemas domóticos, que en sus inicios estaban centrados fundamentalmente en los electrodomésticos y los servicios básicos del hogar. Prueba de ello lo constituye la producción de teléfonos móviles que incluyen aplicaciones para funciones domóticas remotas.

Se suele considerar que la domótica es una especie de disciplina emergente de interface, en la que conjuntamente están implicados arquitectos, ingenieros eléctricos, electrónicos y civiles, programadores de sistemas y diseñadores. En su formación es recurrente que utilicen modelos de vivienda a escala, constituyéndose en un aspecto clave para aplicar y verificar las ventajas y posibilidades de los sistemas. En algunos casos, la formación en domótica dirigida a arquitectos, ingenieros y hasta promotores inmobiliarios, considera en su aplicación práctica las características socio-demográficas emergentes en términos de la estructura de la familia, incluyendo desde el cambio de papel de la mujer en el hogar hasta las condiciones económicas de sus ocupantes.

Por otra parte, también consideramos de especial interés resaltar en este apartado los últimos avances que están teniendo lugar en el campo de la Robótica y, en particular, el uso de ésta en el ámbito doméstico. La palabra robot se utiliza para describir cualquier tipo de máquina que realiza de forma automática tareas humanas, sin que sea necesaria la actuación del hombre una vez que han sido programados. Lo que indican las cifras es un boom en el sector de robots para el hogar, coincidente con la demanda de robots en el sector industrial. Cada vez existirán un mayor número de consumidores que comprarán estos artefactos de última generación para cortar el césped, aspirar suelos, limpiar ventanas y realizar otras tareas domésticas. El desarrollo tecnológico permitirá que estos robots sean cada vez más eficaces y menos aparatosos, aspectos que les harán ganar en popularidad y reducir su coste. (SOLÁNS 2005)

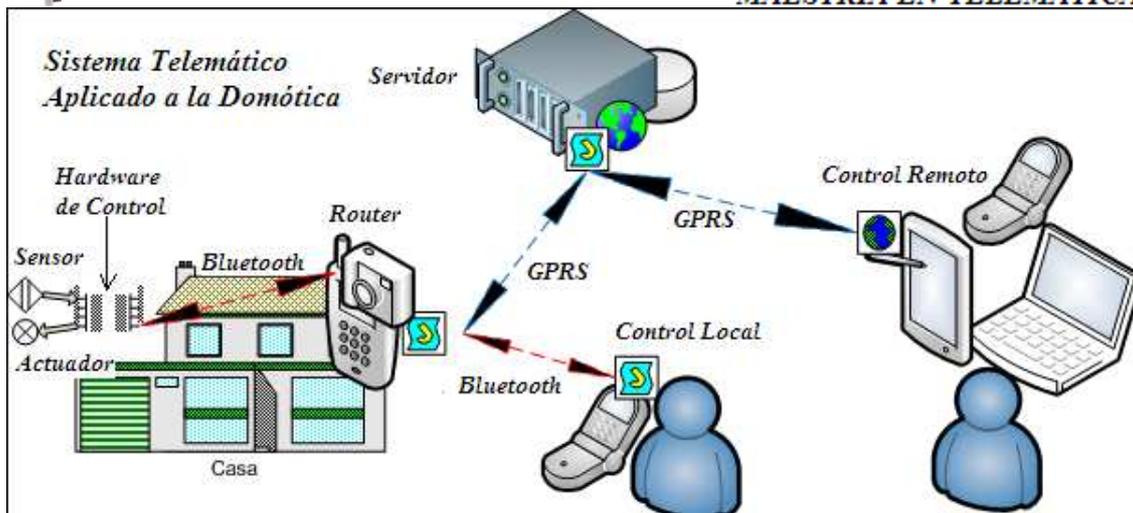


Figura 2-1. Sistema telemático aplicado a la domótica.

En la Figura 2-1, observamos un sistema telemático orientado a brindar confort y seguridad en un domicilio, que es la investigación que se desarrolla a detalle en los siguientes capítulos.

2.2 Celular Digital.

Los sistemas de radiocomunicaciones móviles celulares digitales, trabajan en los 1.8 a 1.9 GHz (PCS de alto nivel) y los 800-900MHz, proporcionando alta movilidad, y una amplia área de cobertura sobre las células macro.

Los sistemas celulares de radio en 800-900 MHz, han evolucionado a la tecnología digital en forma de Sistema Global para Comunicaciones Móviles (GSM) en Europa, PDC (Personal Digital Cellular) en Japón, IS-54 Acceso Múltiple por División de Tiempo (TDMA) y el IS- 95 Acceso Múltiple por División de Código (CDMA) en los EE.UU.

Las capacidades de roaming entre redes celulares ofrecen una amplia cobertura. Las redes celulares se han expandido, con la ampliación de cobertura más allá de las fronteras nacionales.

Estos sistemas de acceso inalámbrico integrado con redes móviles de gran escala, tienen una sofisticada inteligencia permitiendo la movilidad de los usuarios. (GIBSON 1999)



2.3 JAVA ME.

2.3.1 Concepto.

Java ME (Mobile Edition) es la plataforma Java para dispositivos de consumo embebidos, tales como teléfonos móviles, buscapersonas, agendas personales, navegadores de automóviles, dispositivos con tv, y los teléfonos con Internet habilitado. Java ME es una de las tres ediciones de plataformas. Las otras dos ediciones de la plataforma son la plataforma Java, Enterprise Edition (Java EE) para servidores y ordenadores de la empresa y Java Platform, Standard Edition (Java SE) para ordenadores de escritorio. Una tecnología relacionada con la tecnología Java es Card (Trade Mark). La especificación Java Card permite que la tecnología Java se ejecute en tarjetas inteligentes y otros dispositivos con más memoria limitada que un teléfono móvil de gama baja. Estas agrupaciones son necesarias para adaptar la tecnología Java a las diferentes áreas de hoy, de la gran industria de la computación. La Figura 2-2, ilustra la plataforma Java, ediciones y sus mercados de destino.

La plataforma Java ME ofrece el poder y los beneficios de la tecnología Java (Portabilidad del código, la programación orientada a objetos, y un rápido ciclo de desarrollo) para el consumidor y dispositivos embebidos.

El objetivo principal de Java ME es permitir en los dispositivos descargar aplicaciones que aprovechan las capacidades nativas de cada dispositivo.

El Consumidor y sus necesidades abarcan una amplia gama de dispositivos, desde buscapersonas, a tv set-top boxes, que varían ampliamente en memoria, potencia de procesamiento, y capacidades de entrada/salida.

Para hacer frente a esta diversidad, la arquitectura Java ME define configuraciones, perfiles y paquetes opcionales para permitir la modularidad y personalización. (THOMPSON 2008)

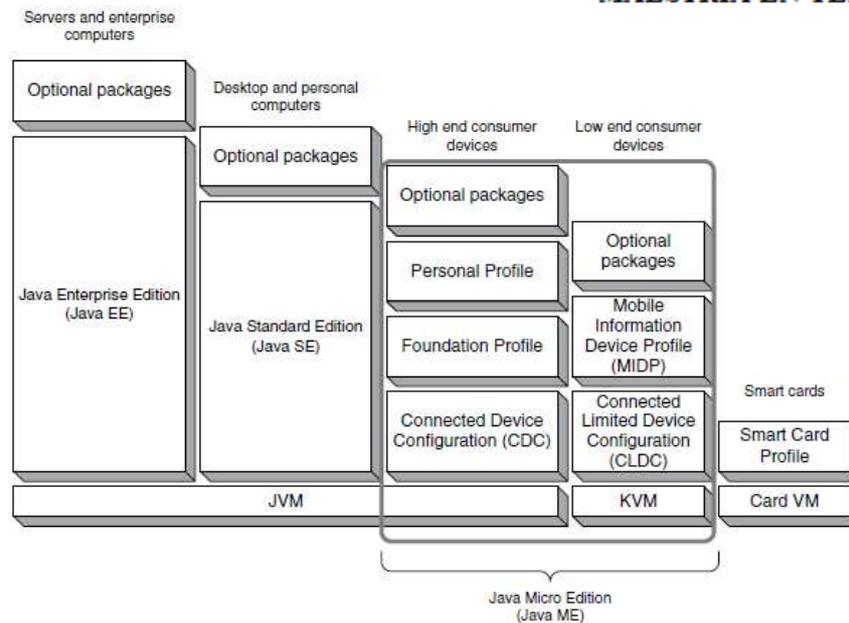


Figura 2-2. Plataformas Java. (THOMPSON 2008)

2.3.2 CLDC.

Una máquina virtual de Java, interpreta los byte codes generados cuando los programas son compilados. Un programa de Java puede correrse en algún dispositivo que tenga una máquina virtual apropiada, y un conjunto apropiado de bibliotecas de clase Java.

Las configuraciones están compuestas de una máquina virtual de Java, y un mínimo conjunto de bibliotecas de clases. En la Figura 2-3, observamos la máquina virtual, que normalmente corre sobre un sistema operativo, que es parte del software del sistema del dispositivo.

La configuración define las mínimas funcionalidades, para una categoría particular o agrupación de dispositivos. Define las capacidades y requerimientos mínimos para la máquina virtual de Java, y bibliotecas de clases disponibles en todos los dispositivos de la misma categoría o agrupación. Actualmente, hay dos configuraciones ME: la CLDC y la CDC. (THOMPSON 2008)

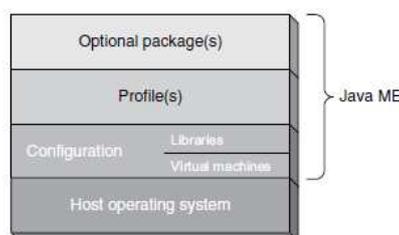


Figura 2-3. Componentes de la Arquitectura Java ME. (THOMPSON 2008)



2.3.3 MIDP.

Las configuraciones usualmente no proporcionan una solución completa. Los perfiles añaden la funcionalidad y las APIs necesarias para completar el entorno de ejecución para una clase de dispositivos. Las configuraciones deberían ser combinadas con los perfiles que definen el más alto nivel de APIs para proporcionar las capacidades para un mercado o industria específica. Esto es posible ya que un único dispositivo soporta varios perfiles. Ejemplos de perfiles son Mobile Information Device Profile (MIDP), Foundation Profile (FP), y Personal Profile (PP). Una aclaración es necesaria: Los perfiles Bluetooth definidos previamente no serán confundidos con los perfiles de Java ME discutidos aquí. Los dos perfiles no están relacionados. Un perfil Bluetooth se refiere a un conjunto de funcionalidades de los protocolos Bluetooth para un caso de uso particular. Los perfiles de Java ME son un conjunto de APIs que amplían las funcionalidades de una configuración Java ME. (THOMPSON 2008).

2.3.4 JSRs.

Para hacernos una idea de lo que está pasando en el mundo J2ME, la Tabla 2-1 muestra algunas de las configuraciones, perfiles y APIs opcionales que están disponibles y en desarrollo. Esta no es una lista completa, para más información, podemos visitar el sitio web de JCP en <http://jcp.org/>.

| Configuraciones | |
|------------------------|--|
| JSR | Nombre |
| 30 | Connected, Limited Device Configuration (CLDC) 1.0 |
| 139 | Connected, Limited Device Configuration (CLDC) 1.1 |
| 36 | Connected Device Configuration 1.0.1 |
| 218 | Connected Device Configuration 1.1 |
| Perfiles | |
| 37 | Mobile Information Device Profile 1.0 |
| 118 | Mobile Information Device Profile 2.0 |
| 75 | PDA Profile 1.0 |
| 46 | Foundation Profile 1.0 |



| | |
|------------------------|--------------------------------|
| 129 | Personal Basis Profile 1.0 |
| 62 | Personal Profile 1.0 |
| 219 | Foundation Profile 1.1 |
| 217 | Personal Basis Profile 1.1 |
| APIs Opcionales | |
| 75 | PDA Optional Packages for J2ME |
| 82 | Java APIs for Bluetooth |
| 135 | Mobile Media API 1.1 |
| 184 | Mobile 3D Graphics |
| 179 | Location API for J2ME |
| 120 | Wireless Messaging API 1.0 |
| 205 | Wireless Messaging API 2.0 |
| 172 | J2ME Web Services APIs |
| 66 | RMI Optional Package |

Tabla 2-1. J2ME Configuraciones, Perfiles, y APIs Opcionales. (KNUDSEN 2005)

2.3.5 Servlets.

Son aquellos, que utilizan un modelo petición-respuesta HTTP de comunicación entre cliente y servidor.

Los servlets extienden la funcionalidad de un servidor, al permitir que éste genere contenido dinámico. Por ejemplo, los servlets pueden generar en forma dinámica documentos XHTML personalizados, ayudar a proporcionar un acceso seguro a un sitio Web, interactuar con bases de datos a beneficio de un cliente y mantener la información de sesión única para cada cliente. Un componente del servidor Web, conocido como contenedor de servlets, ejecuta los servlets e interactúa con ellos. Los paquetes javax.servlet y javax.servlet.http proporcionan las clases e interfaces para definir servlets. El contenedor de servlets recibe peticiones HTTP de un cliente y dirige cada petición al servlet apropiado. El servlet procesa la petición y devuelve una respuesta apropiada al cliente; por lo general en forma de un documento XHTML o XML (Lenguaje de marcado extensible) para mostrarlo en el navegador. XML es un lenguaje que se utiliza para intercambiar datos estructurados en la Web.



Desde el punto de vista arquitectónico, todos los servlets deben implementar a la interfaz Servlet del paquete javax.servlet, la cual asegura que cada servlet se pueda ejecutar en el marco de trabajo proporcionado por el contenedor de servlets. La interfaz Servlet declara métodos que el contenedor de servlets utiliza para administrar el ciclo de vida del servlet. Este ciclo de vida empieza cuando el contenedor de servlets lo carga en memoria; por lo general, en respuesta a la primera petición del servlet. Antes de que el servlet pueda manejar esa petición, el contenedor invoca al método init del servlet, el cual se llama sólo una vez durante el ciclo de vida de un servlet para inicializarlo. Una vez que init termina su ejecución, el servlet está listo para responder a su primera petición. Todas las peticiones se manejan mediante el método service de un servlet, el cual es el método clave para definir la funcionalidad de un servlet. El método service recibe la petición, la procesa y envía una respuesta al cliente. Durante el ciclo de vida de un servlet, se hace una llamada al método service por cada petición. Cada nueva petición se maneja comúnmente en un subproceso de ejecución separado (administrado por el contenedor de servlets), por lo que cada servlet debe ser seguro para los subprocesos. Cuando el contenedor de servlets termina el servlet (por ejemplo, cuando el contenedor de servlets necesita más memoria o cuando se cierra), se hace una llamada al método destroy del servlet para liberar los recursos que éste ocupa. (DEITEL, Paul J. y Harvey M. DEITEL 2008)

2.4 Red GPRS.

2.4.1 Concepto.

GPRS (General Packet Radio System-Sistema General de Radio por Paquetes) es una tecnología para la transmisión de datos en redes de telefonía móvil que utiliza un sistema de conmutación de paquetes en vez del clásico de conmutación de circuitos como se utiliza en el sistema GSM tradicional.

Esta tecnología representa el paso previo a las llamadas tecnologías de tercera generación (3G), por lo que ha sido denominada por algunos como 2.5 G ya que se trata de la generación intermedia entre la actual GSM (2 G) y la red UMTS.

GPRS, es una red que se superpone a la actual red GSM, teniendo ambas en común el interfaz radio, pero diferenciándose de ella en que es una red que como se ha dicho anteriormente es exclusivamente para datos transmitidos mediante conmutación de paquetes. (ESPINOSA, Técnico en Telecomunicaciones, Tomo III 2002)



2.4.2 Características Generales del GPRS.

GPRS, nace con el objetivo de proporcionar comunicaciones de datos móviles con gran eficiencia. Para ello utiliza cuatro mecanismos:

Empleo de esquemas de codificación mejorados respecto a los del GSM y de naturaleza dinámica: en función de las condiciones imperantes en el medio radio se utiliza un esquema u otro. Se puede conseguir una velocidad máxima de 21,4 Kb/s por intervalo. Posibilidad de asignación de varios intervalos de tiempo a una comunicación de datos con reserva independiente de los intervalos para los enlaces ascendente y descendente. De este modo, usando los 8 TS (time slots) puede alcanzarse una tasa de hasta 170 Kb/s y el sistema puede acomodarse al carácter asimétrico de muchas comunicaciones de datos.

Estas cifras deben interpretarse como valores máximos del caudal ya que dependiendo de las condiciones del enlace de radio, de las retransmisiones de paquetes necesarios para garantizar la integridad de éstos, así como de la eventual compartición de los recursos, el caudal medio disponible por un usuario puede ser bastante inferior al valor máximo.

Posibilidad de compartición de recursos radio entre varios usuarios, mediante multiplexación dinámica.

Utilización de la conmutación de paquetes, tanto en la red como en el acceso radio.

Como consecuencia de estas funcionalidades el GPRS puede proporcionar a los usuarios:

Conectividad desde el teléfono móvil con redes de paquetes (IP, X.25). GPRS utiliza pasarelas, de forma que estas redes contemplan el móvil como un terminal IP más quedando oculta para ellas la red GPRS.

Un servicio móvil de datos de gran eficiencia y con un caudal idóneo para soportar aplicaciones basadas en X.25 y TCP/IP, abriendo el camino a la Internet Móvil.

Conectividad permanente (always on) con la red, eliminándose los tiempos de establecimiento de las llamadas.

Mejora sustancial del servicio de mensajería, superando la limitación de 160 caracteres del SMS/GSM y permitiendo servicios de mensajería multimedia MMS con mensajes de voz, texto, imágenes estáticas y video.

Posibilidad de elección entre varios parámetros de calidad de servicio.



Facturación de los servicios por volumen de información intercambiada y no por tiempo. Ello es muy ventajoso dado el carácter intermitente, en ráfagas, del tráfico de datos.

El aumento del caudal de transmisión, uno de los objetivos primordiales del GPRS tiene un límite impuesto por el sistema de modulación subyacente que es el mismo que para GSM. Para ampliar este límite se ha desarrollado otra variante del GSM que es la tecnología EDGE en la que se utiliza una modulación 8PSK. La combinación con GPRS: EGPRS permite alcanzar la velocidad teórica de 384 kb/s que ya pertenece a la Tercera Generación. (HERNADO 2004)

2.5 Tecnología Inalámbrica Bluetooth.

La tecnología inalámbrica Bluetooth es una especificación abierta, de corto alcance, de baja potencia, y bajo costo, es la tecnología de radio para comunicaciones inalámbricas ad-hoc de voz y datos en cualquier parte del mundo. Examinemos cada uno de estos atributos.

- Una especificación abierta significa que la especificación esté disponible al público y libre de derechos.
- En ésta tecnología de radio de corto alcance, los dispositivos pueden comunicarse a través del aire mediante ondas de radio a una distancia de 10 metros (m). Con la más alta transmisión de energía aumenta el alcance a aproximadamente 100 m.
- Debido a que las comunicaciones son en un rango corto, las radios son de baja potencia, y son adecuados para los dispositivos portátiles que funcionan con baterías.
- La tecnología inalámbrica Bluetooth soporta tanto voz, como datos, permitiendo a los dispositivos comunicar ambos tipos de contenidos.
- La tecnología inalámbrica Bluetooth funciona en cualquier parte del mundo ya que funciona a 2,4 GHz, disponible a nivel mundial en la banda no licenciada (ICM), industrial, científica y médica. En la Tabla 2-2, tenemos una comparativa de tres tecnologías inalámbricas, para comprender mejor las características de la comunicación Bluetooth.

Debido a que la banda de frecuencia ICM está disponible para uso general de aplicaciones ICM, muchos otros dispositivos (por ejemplo, WiFi, teléfonos inalámbricos, hornos de microondas) operan en esta banda. La tecnología inalámbrica



Bluetooth está diseñada para ser muy robusta frente a interferencias de otros dispositivos. (THOMPSON 2008)

| Características y función | IrDA | Wireless LAN | Comunicación Bluetooth |
|----------------------------------|---|---|-------------------------------|
| <i>Tipo de conexión</i> | Infrarrojo, haz estrecho, línea de vista. | Espectro ensanchado, esférico | Espectro ensanchado, esférico |
| <i>Espectro</i> | Óptico 850 - 900 nm | RF 2.4 GHz(5GHz para 802.11 a/n) | RF 2.4 GHz |
| <i>Potencia de Transmisión</i> | 40 a 500 mW/Sr | 100 mW | 10-100 mW |
| <i>Máxima tasa de datos</i> | 9600 bps-16Mbps (muy raro) | 11 Mbps (54Mbps para 802.11a y 802.11g) | 3 Mbps |
| <i>Alcance</i> | 1 m | 100 m | 10-100 m |
| <i>Dispositivos soportados</i> | 2 | Se conecta a través de un punto de acceso | 8 (activos), 200 (pasivos) |
| <i>Canales de voz</i> | No | No | Si |
| <i>Direccionamiento</i> | 32-bit ID físico | 48-bit MAC | 48-bit MAC |

Tabla 2-2. Comparación de Tecnologías de Comunicación Inalámbrica. (THOMPSON 2008)



2.6 PIC16F877A.

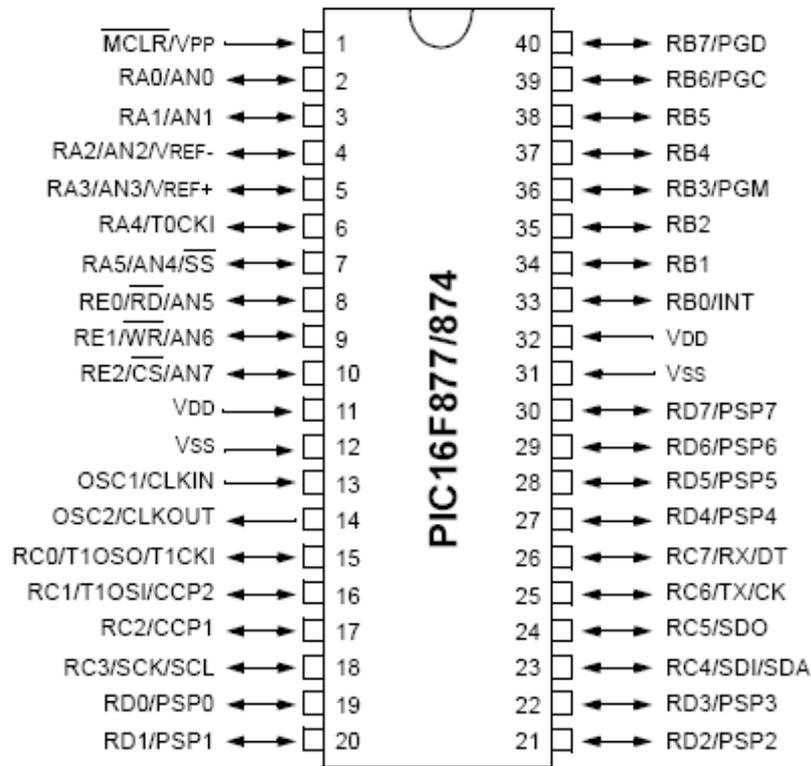


Figura 2-4. PIC16F877A. (Microchip 2003)

En la Figura 2-4, observamos los nombres de los pines del PIC16F877A. El PIC16F877A, es un microcontrolador de 40 pines y, algunas de sus características principales son:

Empaquetado: DIP 40 pines.

Memoria FLASH: 8K.

Memoria RAM: 368 bytes.

Memoria EEPROM: 256 bytes.

Convertidor Analógico Digital de 10 bits de resolución con 8 canales.

Dos comparadores analógicos.

USART (Universal Synchronous Asynchronous Receiver Transmitter)

Se ha seleccionado el PIC 16F877A por la facilidad para comunicarse con otros dispositivos (USART), por el número de puertos, y por la facilidad para encontrarlo en el mercado.



2.7 MicroCode Studio 3.0.0.5

MicroCode Studio es un poderoso entorno de desarrollo integrado visual (IDE) con capacidades de depuración en el circuito (ICD), diseñado específicamente por microEngineering Labs PICBASIC y compilador PICBASIC PRO.

Posee las siguientes características principales:

- Resaltado completo de sintaxis de su código fuente.
- Salto rápido para incluir archivos, símbolos, defines, variables y etiquetas usando la ventana del explorador de código.
- Identifica y corrige errores de compilación y ensamblado.
- Permite ver la salida serial de tu microcontrolador.
- Ayuda sensitiva, contexto basado en palabras clave.
- Soporte para MPASM.

MicroCode Studio incluye ahora EasyHID Wizard, una herramienta de generación de código libre que permite a un usuario ejecutar rápidamente la comunicación bidireccional entre un microcontrolador incrustado PIC™ y un PC. (Mechanics s.f.)

2.8 Simulador Proteus 7.6

Software de diseño de PCB profesional con auto ruteo basado en integración de formas.

Algunas características son:

- Captura esquemática de las características completas.
- Autorruteo basado en formas estándares de clases.
- Reglas de diseño altamente configurables.
- Simulador interactivo de circuitos SPICE.
- Soporte extendido para planos de potencia.
- Salidas estandarizadas para la industria CAD/CAM & ODB++.
- Visor 3D integrado. (Electronics s.f.)

2.9 Módulo Bluetooth RN-41.

El RN-41 es un módulo de tamaño pequeño, de baja potencia, es una muy económica solución Bluetooth para fabricantes de equipo original, que permite añadir capacidad de conexión inalámbrica a sus productos. El RN41, soporta múltiples protocolos de interfaz, es fácil de diseñar, y totalmente certificado, convirtiéndolo en una completa



solución Bluetooth incorporada. Por soportar la especificación Bluetooth EDR (Tasa de Datos Mejorada), el RN-41 ofrece tres veces la velocidad de datos que los dispositivos Bluetooth v1.2. Los diseñadores pueden personalizar sus aplicaciones usando la interfaz externa para acceder a 8Mbits de memoria flash. El RN-41 es el módulo perfecto para los desarrolladores que quieren añadir capacidad inalámbrica a sus productos, pero no puede permitirse el lujo de gastar mucho tiempo y dinero en el desarrollo de hardware y software específico para Bluetooth. (RovingNetworks s.f.) El Anexo A-2, muestra a detalle la hoja de especificaciones técnicas del fabricante.

2.10 Sensor de Movimiento.



Figura 2-5 Sensor de Movimiento. (BricoGeek.com s.f.)

En la Figura 2-5, observamos, un simple pero poderoso sensor PIR detecta el movimiento de cualquier objeto o persona en toda una habitación. Simplemente conectado y esperando unos 2 segundos estará listo. Cuando detecte algún movimiento, el pin de "alarma" pasará a nivel bajo.

Importante:

El conexionado, es el siguiente:

Cable rojo: VCC (5 a 12V)

Marrón: GND

Cable negro: Pin "Alarma" en colector abierto

Dado que el pin de alarma es de colector abierto, se necesita una simple resistencia Pull Up.

En el Anexo A-4, se encuentran más características técnicas del dispositivo electrónico.



2.11 Actuadores.

2.11.1 Luminarias

Los tipos principales de luminaria son la fluorescente y la incandescente:

La lámpara fluorescente, también denominada tubo fluorescente, es una lámpara de vapor de mercurio a baja presión, utilizada para la iluminación doméstica e industrial. Su gran ventaja frente a otro tipo de lámparas, como las incandescentes, es su eficiencia energética.

Está formada por un tubo o bulbo fino de vidrio revestido interiormente con una sustancia que contiene fósforo y otros elementos que emiten luz al recibir una radiación ultravioleta de onda corta. El tubo contiene una pequeña cantidad de vapor de mercurio y un gas inerte, habitualmente argón o neón, sometidos a una presión ligeramente inferior a la presión atmosférica. Asimismo, en los extremos del tubo existen dos filamentos hechos de tungsteno. (Luminaria fluorescente s.f.)

La lámpara incandescente; llamada también bombilla, ampolleta, lamparita, foco, bombillo o bombita de luz; es un dispositivo que produce luz mediante el calentamiento por efecto Joule de un filamento metálico, hasta ponerlo al rojo blanco, mediante el paso de corriente eléctrica. En la actualidad, técnicamente son muy ineficientes ya que el 90% de la electricidad que utilizan la transforman en calor. (Lámpara incandescente s.f.)

Dos métodos de Control de la Iluminación con Domótica bastante usados son:

Simulación de Presencia._ La simulación de presencia tiene como objetivo hacer parecer que la casa está habitada aunque está vacía. La iluminación puede ser utilizada (con o sin otros elementos integrados en el control del sistema de domótica) para la simulación de presencia en la vivienda, encendiendo y apagando la iluminación ciertas horas del día, de forma programada, aleatoria, o de unas rutinas aprendidas por el sistema de domótica.

Control Manual._ El control manual de la iluminación se puede realizar a través de una gran variedad de interfaces, como pulsadores de pared, mandos a distancia, Web e incluso por voz). Es preciso indicar que, aunque se integra el control de la iluminación en un sistema de domótica, normalmente se debería garantizar la posibilidad de controlar la iluminación mediante interruptores tradicionales. (Iluminación s.f.)

2.11.2 Cerradura Eléctrica

La cerradura eléctrica está presente en muchos sistemas de porteros y alarmas, así como también en muchos sistemas domóticos.



La cerradura eléctrica para puertas mostrada en la Figura 2-5, es un electroimán con un acople mecánico que permite la apertura de la puerta en la que se encuentra instalada.



Figura 2-6 Cerradura Eléctrica FE 789. (Cerradura eléctrica FE-789 Llave Plain Key (Multipunto) s.f.)

Características:

Cerradura eléctrica inoxidable con perfil plain key (llave multipunto);

Modelos de sobreponer de alta eficiencia;

Pestillo y roldana impulsora en latón;

Tirador revestido para evitar choques y cortes;

Instalación derecha/izquierda, apertura para dentro o para fuera;

Memoria mecánica, desbloquea en el primer impulso eléctrico;

Alimentación 12 VCA / 0,8 A;

Aplicación en puertas y portones. (Cerradura eléctrica FE-789 Llave Plain Key (Multipunto) s.f.)

2.11.3 Puerta Automática

La Puerta de Garaje, es hoy en día casi siempre motorizada y controlada por un mando a distancia. Pero para crear mayor confort y seguridad, la apertura de la puerta del garaje puede además ser integrada con el sistema de domótica. Por un lado permite la apertura de la puerta del garaje y también llegar a ser combinada con otras actuaciones como el encendido de la iluminación, exterior e interior, la desconexión del sistema de seguridad, etc. Y a la salida automáticamente apagar las luces, bajar las persianas y activar la alarma de la casa. (Puertas y ventanas s.f.)



UNIVERSIDAD DE CUENCA
FACULTAD DE INGENIERÍA
MAESTRÍA EN TELEMÁTICA

Para todo tipo de puerta (basculante desbordante y no desbordante, seccional, plegable y batiente), se distingue por un seguro contra intrusión que previene eventuales tentativas de robo.

CAPITULO 3: DISEÑO E IMPLEMENTACIÓN DEL HARDWARE.

3.1 Análisis del problema

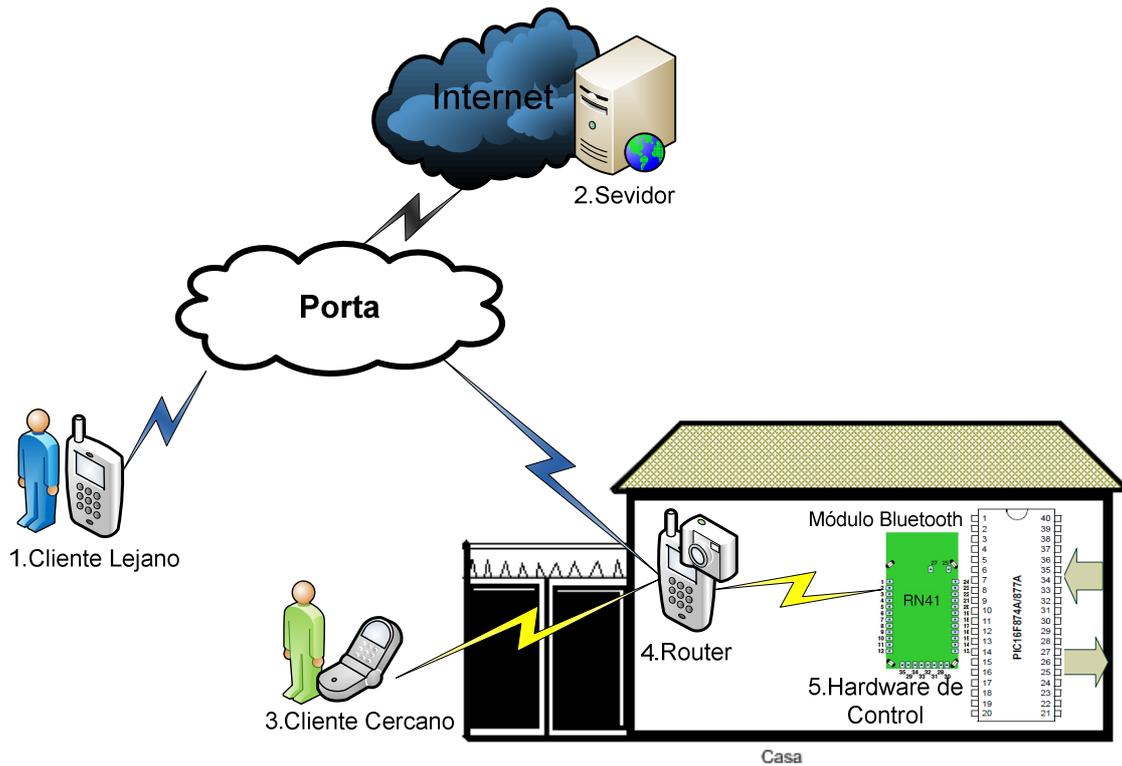


Figura 3-1 Análisis del problema.

En la Figura 3-1, observamos los componentes del sistema de telemedición y telecontrol, que son los siguientes:

- 1. Cliente Lejano**, el cual interactúa con el sistema a través de un navegador de un móvil, tan sofisticado al de un computador normal, ya que cuenta con un navegador que soporta HTML. El cliente lejano interactúa desde un móvil (o portátil) hasta un servlet en el servidor Apache-Tomcat, permitiendo de esta manera el envío de telemandos, la recepción del estado de los sensores, y el registro de usuarios.
- 2. Servidor**, es un servidor Apache-Tomcat, el cual almacena y ejecuta la aplicación web desarrollada en Java (servlet).

3. Cliente Cercano, el cual interactúa con el sistema mediante una aplicación J2ME, de esta forma el Cliente Cercano, envía y recibe comandos, así como también tiene la opción de registrar nuevos usuarios.

4. Router, que es el encargado de enrutar los comandos bidireccionales que llegan tanto del hardware como del Cliente Cercano hacia y desde la aplicación web en el Servidor; y, además se encarga del envío de imágenes hacia el Servidor.

5. Hardware de Control, permite unir el Router con los sensores y actuadores a través de un enlace Bluetooth, en este tercer capítulo vamos a detallar a fondo el diseño e implementación del mismo.

3.2 Diagrama de bloques del hardware del sistema

En el hardware de control, observamos principalmente los siguientes componentes: un módulo Bluetooth RN-41, un microcontrolador PIC16F877A; y, como entradas y salidas del prototipo 4 pulsadores (sensores) y 14 diodos LED (actuadores).

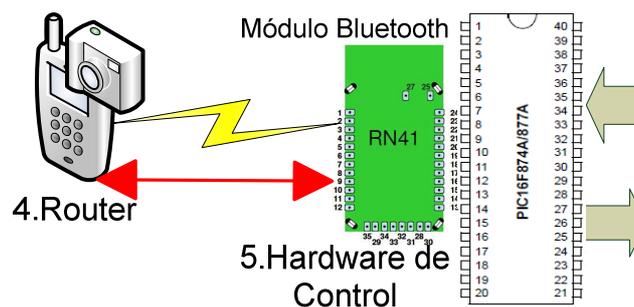


Figura 3-2 Análisis del hardware que interactuará con el router.

El módulo Bluetooth RN-41, es de clase 1, es decir tiene un alcance de 100 m. (contra otro dispositivo de la misma clase), y además se lo ha configurado vía hardware para que trabaje a 9600 baudios en la comunicación bidireccional con el PIC.

En la Figura 3-2, observamos dos flechas que representan las 4 entradas de sensores y las 14 salidas de los actuadores.



3.3 Diagrama de flujo del firmware del PIC.

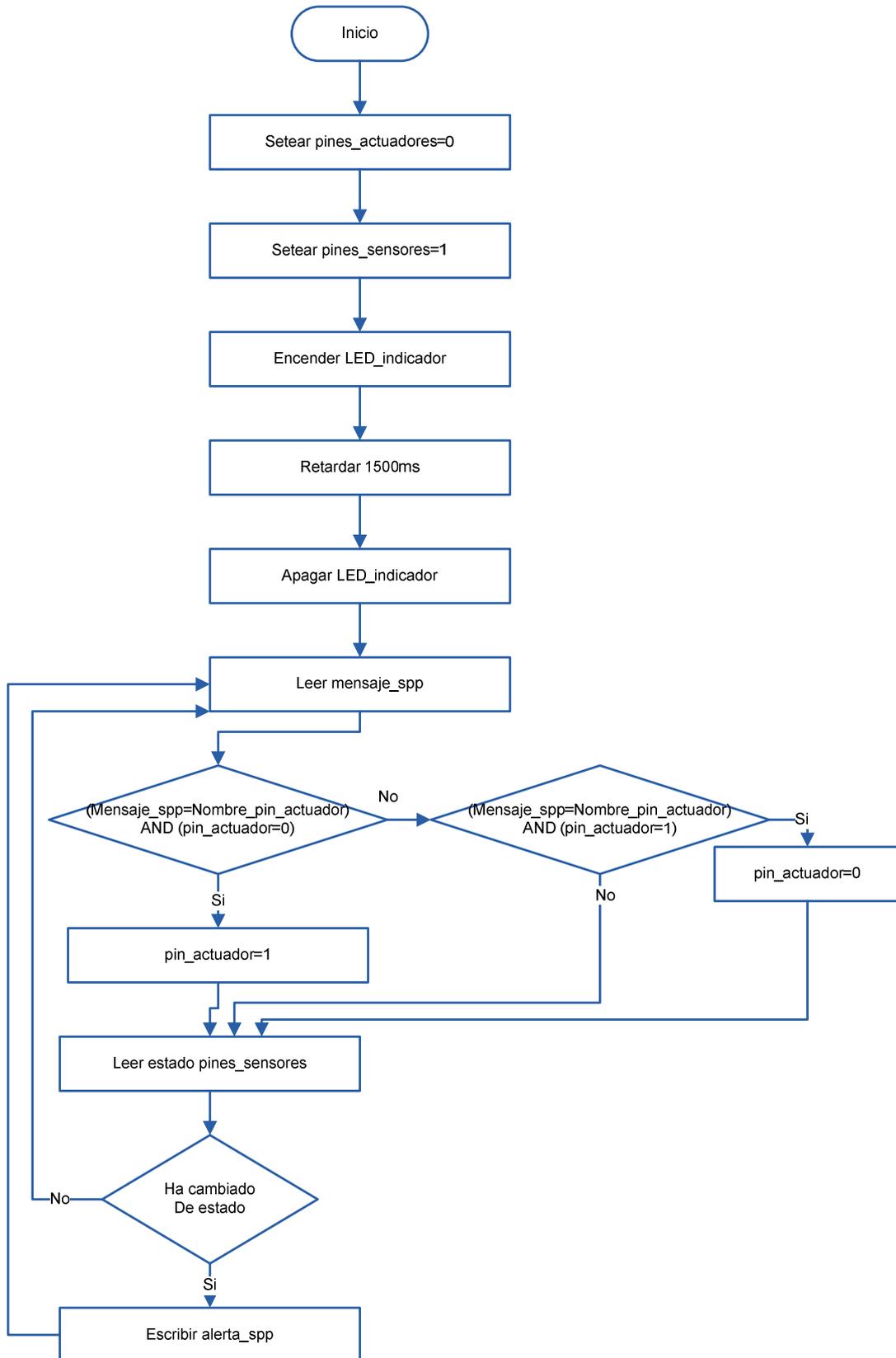


Figura 3-3 Diagrama de flujo para el PIC.



En el romboide de decisión de la derecha de la Figura 3-3, el "Nombre_pin_actuador", puede ser uno de los 14 nombres destinados para los dispositivos actuadores: Pg, Ps, Pc, Pe, Pt, Pd, Ls, Lc, Le, Ld, R1, R2, R3, F1. Y "alerta_spp", son los comandos enviados desde el hardware al "router", y que son S1, S2, S3 y S4.

3.4 Codificación en MicroCode Studio.

En el anexo A-5, observamos a detalle del firmware del PIC del Hardware de Control. Las instrucciones principales usadas para la codificación son las siguientes:

- DEFINE se puede utilizar para cambiar el valor predefinido del oscilador, los pines de DEBUG, la velocidad de transmisión y las ubicaciones de los pines para una LCD, entre otras cosas.

DEFINE OSC 4

Es un ejemplo de cómo puede configurar el cristal que se usará. Las velocidades del oscilador en MHz son: 3(3.58) 4 8 10 12 16 20 24 25 32 33 40 48 64.

- INCLUDE, nos permite añadir archivos fuente BASIC en nuestros programas de PBP. Estos archivos pueden ser incluidos en el código fuente exactamente donde se coloca el INCLUDE, por ejemplo:

INCLUDE "modedefs.bas"

Esto incluye los modos de comunicación serial asíncrona, a mi actual codificación.

- VAR BYTE, con número de bits de 8, y un rango 0 to 255, para nuestro caso en particular, tenemos:

MENSAJE2 VAR Byte[4]

En esta línea de código se almacena los comandos entrantes dentro del arreglo de bytes "MENSAJE2", de 4 posiciones de memoria, en realidad para nuestro caso tranquilamente se usaría de 2 posiciones de memoria nada más, ya que los comandos constan de dos letras (Pg, Pc, Ps,).

- SERIN2 PORTC.7,84,[skip(0),str mensaje2\2] .

Leo los datos que le llegan al PIC, a través del pin PORTC.7. En tanto que el 84, es un número calculado de la siguiente manera:

$(1000000 / \text{baud}) - 20$, para nuestro caso en particular, tenemos la siguiente ecuación:

$(1000000 / 9600) - 20 = 84.17 \rightarrow 84$



El skip(0), afirma que no se saltará ningún caracter a partir de la cadena almacenada.

Str mensaje2\2, es una variable de tipo cadena, la misma que solamente tomará dos caracteres de toda la cadena entrante.

- If (mensaje2[0]="P") and (mensaje2[1]="g") AND (PORTB.1=0)THEN Pg:

Esta línea de código, permite que si la respuesta a las tres preguntas es verdadera, se ejecute la subrutina Pg, que permitirá activar el pin correspondiente a la puerta del garage. Dicho en otras palabras si el comando almacenado en mensaje2 es igual a Pg y además el pin correspondiente a la puerta del garage está desactivado, entonces ejecutar la subrutina "Pg:", que permitirá activar dicho pin.

- Serout2 PORTC.6,84,["S1",13]

Esta línea de código, envía datos al celular denominado "router" desde el hardware. Lo que hace exactamente es transmitir "S1" seguido de un enter, que significa una alerta del sensor 1 (de los cuatro sensores que posee el hardware). Además el 84 indica la velocidad de transmisión y obviamente serout2, indica que es una función de PBP que permite el envío de cadenas de caracteres, en tanto que serout, sólo permite el envío de caracteres individuales.

3.5 Interface de comunicación entre el módulo Bluetooth y el PIC.

Para comprender más a detalle el RN-41, en los Anexos A-2 y A-3 podemos observar el datasheet y el pinout del módulo Bluetooth respectivamente. La interface de comunicación entre el módulo Bluetooth y el PIC, es una comunicación serial a 9600 bps, la misma que permite el envío y recepción de comandos, los cuales enviarán ordenes a los actuadores, así como obtendrán el estado de los actuadores, informando si alguno fue activado.

Por defecto el RN-41 trabaja a 115200bps, por lo cual se requirió configurar vía hardware la velocidad requerida que es de 9600bps, para lo cual se recurrió al *datasheet* (ver Anexo A-3), en el cual indica que el pin 4 puesto en un estado activo, permitirá una comunicación a 9600bps con el PIC. Vale recalcar que se configura la comunicación del RN-41 a 9600 bps, ya que es la que también está configurada en el firmware del PIC, y además es la más común, y hasta por defecto disponible en la hiperterminal; por lo cual, facilita las pruebas del hardware.

3.6 Simulación del Hardware en Proteus 7.6.

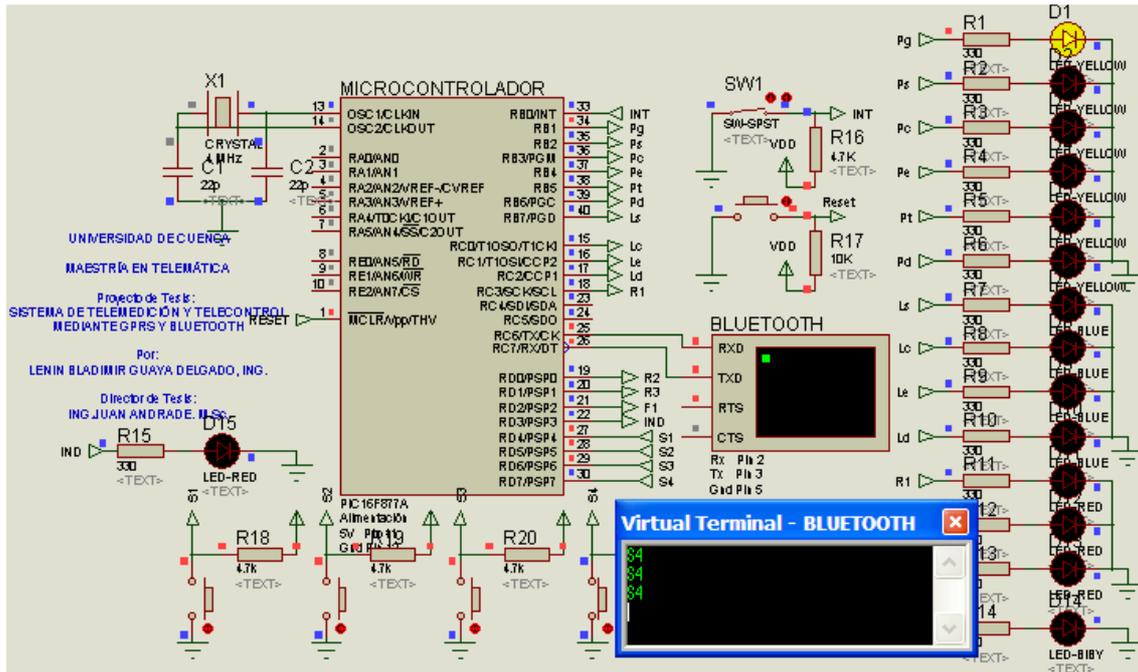


Figura 3-4 Simulación del hardware en Proteus.

En la simulación de la Figura 3-4, es en donde se verifica que el algoritmo implementado está funcionando según los requerimientos, y en caso de ser necesario se podría cambiar funcionalidades del firmware; o talvés aumentar o disminuir componentes al diseño.

La Figura 3-4 muestra claramente los 4 sensores representados con 4 botones pulsadores, que permiten simular el mismo cambio de estado que nos daría el sensor, así lo demuestra el datasheet del sensor que se encuentra en el Anexo A-4.

Además observamos los 14 diodos LED, que representan a los actuadores, éstos permiten ser encendidos y apagados vía comandos Bluetooth. En la gráfica vemos que se ha activado el primer diodo correspondiente a la puerta del garage vía comandos Bluetooth enviados desde la consola de negro (virtual terminal).

También observamos una terminal virtual que simula exactamente el comportamiento del RN-41. Lo que se está probando es que mientras mantengo pulsado el S4 (activandolo), el dato que envía es "S4", con lo cual verificamos el acertado funcionamiento del hardware.

En el sector superior derecho de la figura mostrada, encontramos un interruptor y un pulsador, el interruptor nos permite cambiar de modo, para una futura implementación



de alguna funcionalidad extra a nivel de hardware; el pulsador (Reset), en cambio nos permite reiniciar la ejecución del firmware del PIC.

Ahora centrémonos en los pines de transmisión y recepción tanto del PIC como del RN-41, y teóricamente solo tendrían una entrada y una salida serial, así lo deja ver el datasheet, pero vía instrucciones de PBP, podemos entablar varias comunicaciones seriales con tan solo explicar en la función `serin2` (o `serout2`), que el pin, es uno de los disponibles para datos; en el caso del 16F877A, tenemos 5 puertos de datos.

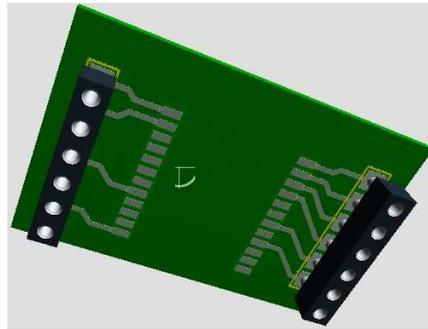


Figura 3-5 Diseño del PCB para el RN-41.

Ares Professional de Proteus, nos permite obtener un anticipado de lo que será el PCB que se está diseñando, en la Figura 3-5, observamos que se ha diseñado un PCB para ubicar adecuadamente al RN-41, el RN-41 es un componente bastante delicado, por lo que es recomendable ubicarlo correctamente y con su respectivo circuito de alimentación a 3.3 VCD.



Figura 3-6 Diseño del PCB del hardware.

Es una excelente ayuda, ver anticipadamente como está la distribución de los componentes de todo el hardware (ver Figura 3-6), porque podemos planificar la forma, tamaño y particularidades de la caja de protección.

3.7 Implementación.

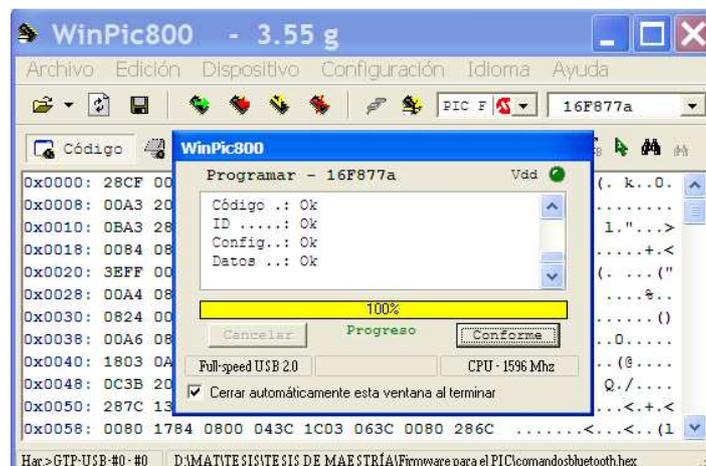


Figura 3-7 Cargando comandosbluetooth.hex en el PIC (1).



Lo que observamos en la Figura 3-7, es la grabación exitosa del programa comandosbluetooth.hex desarrollado en PBP dentro del PIC16F877A.

En la esquina superior derecha observamos el nombre del microcontrolador que es detectado por la aplicación Winpic800 3.55g.

El Winpic800 además tiene funcionalidades para detección del hardware, lectura, escritura, verificación, borrado, permite indicar los datos a grabar en la EEPROM, etc.

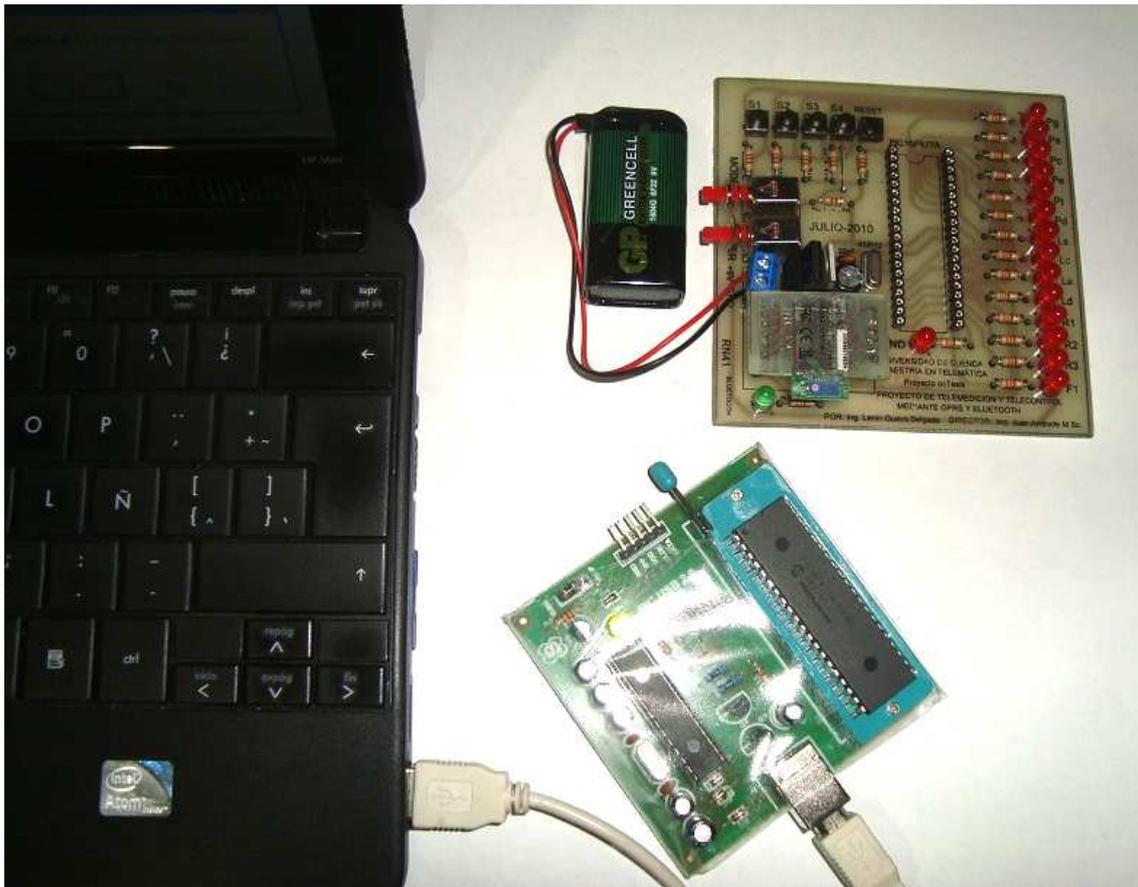


Figura 3-8 Cargando comandosbluetooth.hex en el PIC (2).

En la Figura 3-8, observamos el grabador universal de microcontroladores PIC, la familia más difundida por su versatilidad y abundante bibliografía.

Entonces primeramente grabamos el PIC, y luego lo colocamos en el zócalo del hardware en desarrollo.

Además en la gráfica se puede ver a detalle otra característica del PIC, que tiene un puerto adicional de comunicación ICSP, el mismo que también nos permite grabar PICs, pero más complejos.

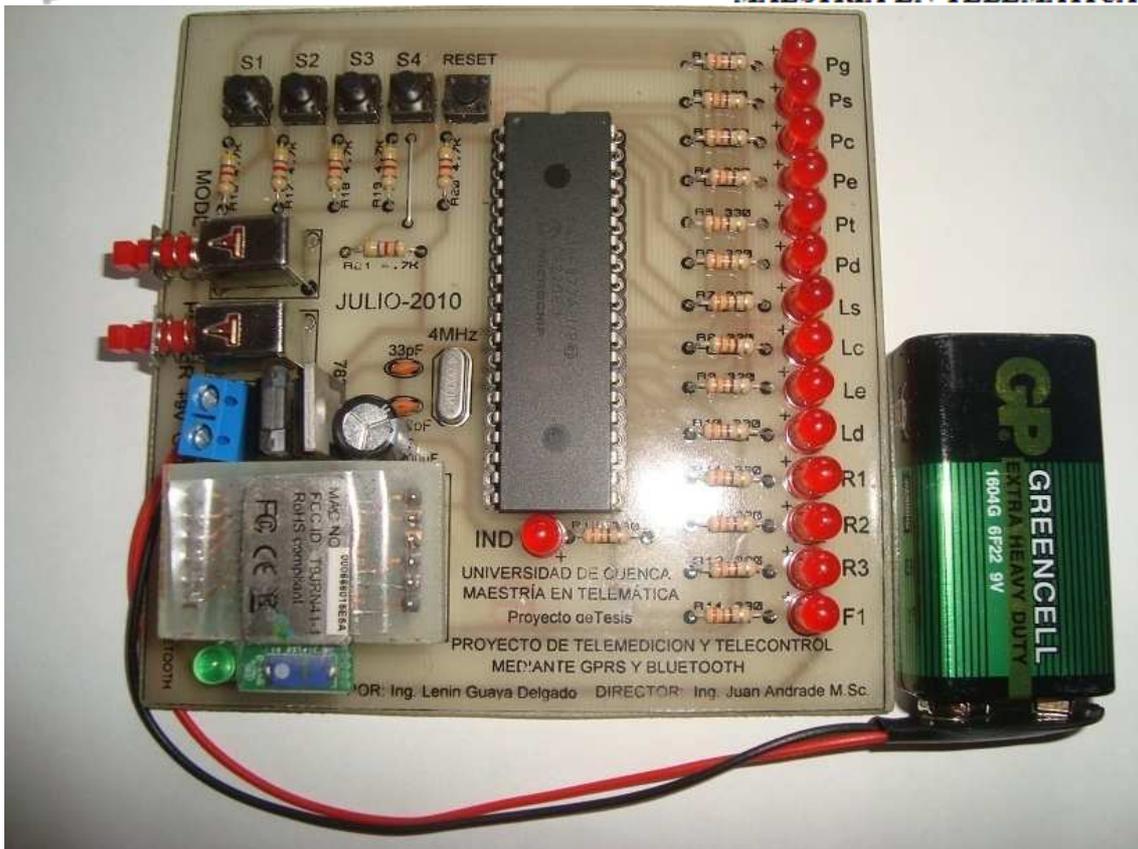


Figura 3-9 Hardware del sistema (frontal).

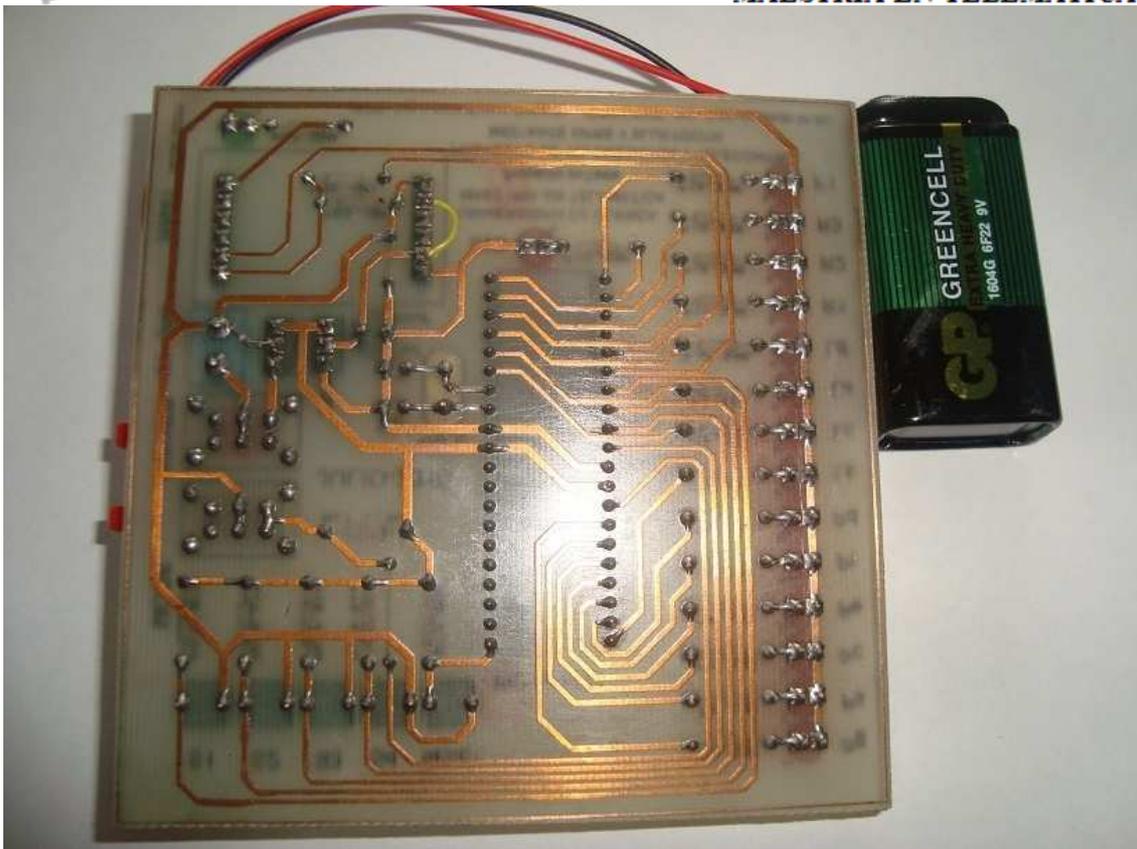


Figura 3-10 Hardware del sistema (reverso).

En las Figuras 3-9 y 3-10 observamos el hardware de control que interactúa con el móvil celular denominado "router" dentro del domicilio.

3.8 Probando el Hardware.

Luego de implementado el PCB, y ser soldados correctamente los componentes que conforman el hardware, es necesario limpiar bien la fibra de vidrio, porque la pasta de soldar nos puede dar falsas conexiones que afectan al normal funcionamiento del hardware, luego de ello mediante un multímetro probamos continuidad en busca de ruteo defectuoso sobre la fibra de vidrio; y, finalmente conectamos la batería de 9 VCD, y comprobamos que en realidad funciona.

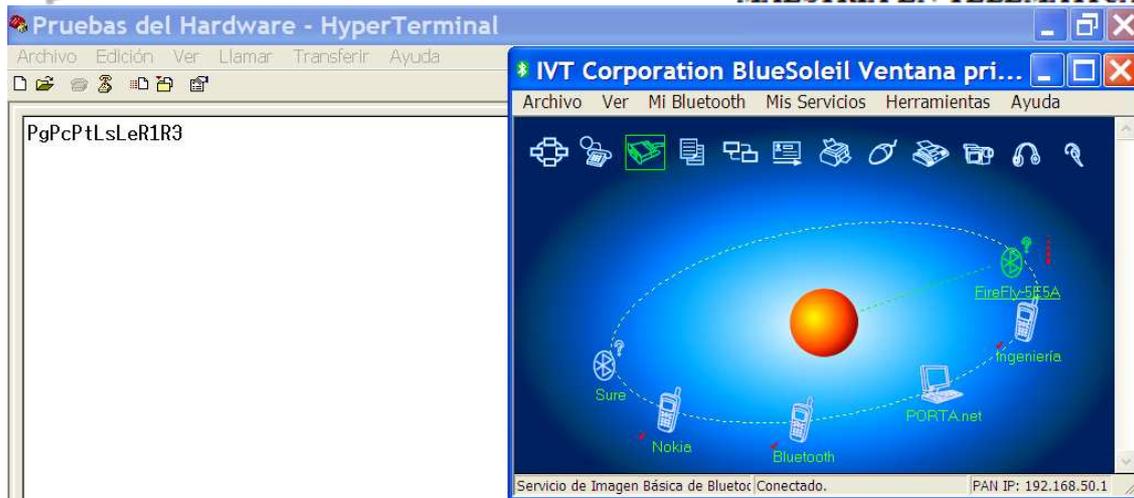


Figura 3-11 Probando el hardware con la hiperteminal.

La Figura 3-11, muestra las pruebas de funcionamiento con ayuda de la Hiperterminal de Windows XP.

En la parte derecha de la figura anterior vemos los dispositivos encontrados, y particularmente de observa la disponibilidad de una comunicación spp para el dispositivo FireFly-5E5A, y además el enlace establecido en línea entrecortada.

En la parte superior izquierda tenemos los siguientes comandos que se han ingresado en la hiperteminal: Pg, Pc, Pt, Ls, Le, R1, R3, comandos que son mostrados en la hiperteminal ya que en Propiedades de la conexión, dentro de configuración ASCII, se ha activado eco de los caracteres escritos localmente. Con los comandos enviados desde la hiperteminal vía bluetooth hasta el hardware estamos dando la orden de que se active los pines correspondientes a Puerta garage, Puerta cocina, Puerta trasera, Lámpara sala, Lámpara estudio, Reflector1 y Reflector3; el resultado de activar estos pines lo vemos en la figura que a continuación se muestra.

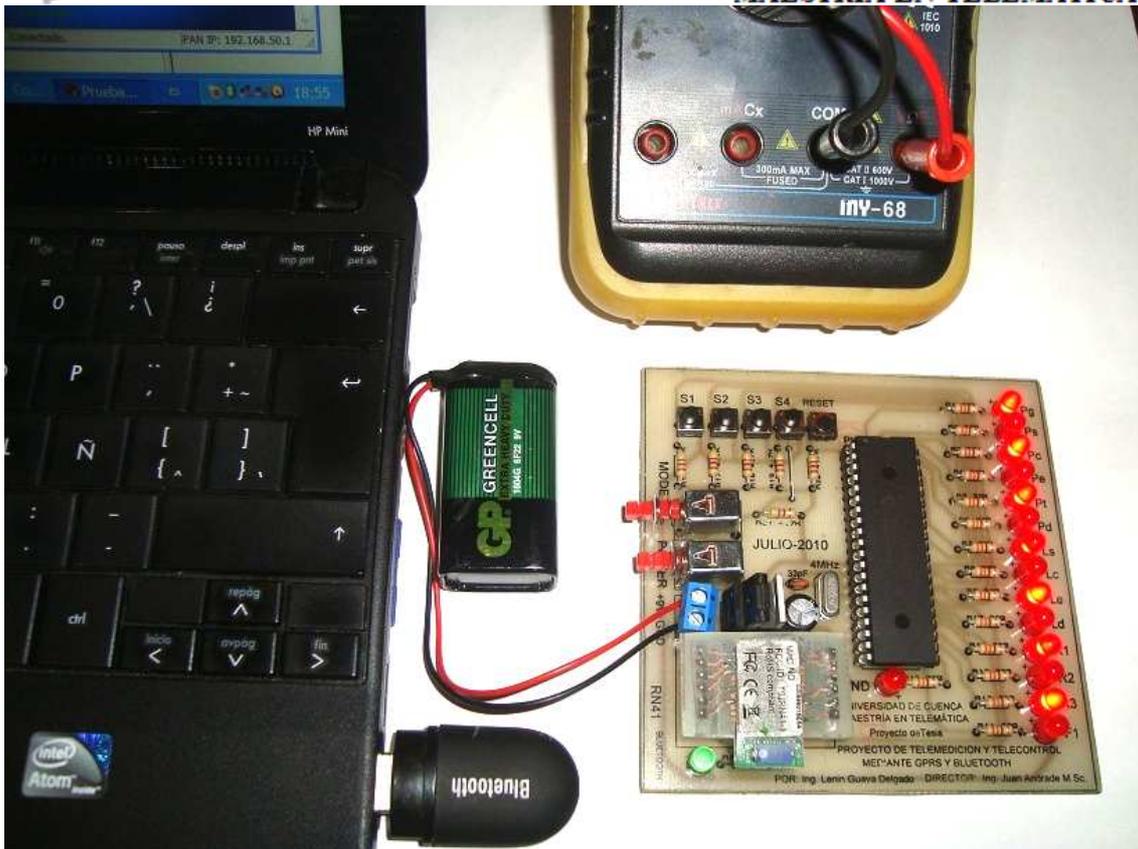


Figura 3-12 Probando salidas para actuadores.

El soporte hardware para poder interpretar los comandos que se envían desde la hiperterminal mostrada en la Figura 3-11, es el que se muestra en la Figura 3-12; aquí vemos el módulo Bluetooth conectado al puerto USB, y los diodos LED encendidos de acuerdo a los comandos ingresados en la hiperterminal.



Figura 3-13 Probando entradas para sensores.

En la Figura 3-13 podemos observar la prueba de recepción de alertas desde el hardware.

Enviamos "xx", y el hardware devuelve "S1", luego enviamos "xx" y recibimos "S2", los datos se los recibe de esa manera puesto que se ha activado las opciones de eco de los caracteres escritos localmente, y agregar avance de línea al final de cada línea recibida. Enviamos "xx", ya que el hardware cada vez que recibe 2 caracteres inicia una nueva interacción, y no importa en realidad que caracteres sean, simplemente espera dos caracteres (diferentes de un comando válido por ejemplo: Pg, Pc, Ls....).

Vale recalcar que se usa la hiperterminal únicamente para pruebas, en realidad la comunicación es entre el hardware y el celular "router".



CAPÍTULO 4: ANÁLISIS DEL PROBLEMA DEL SOFTWARE DEL SISTEMA.

4.1 Análisis del Problema.

Para poder conocer a fondo el problema, vamos a detallar las funcionalidades que el sistema debe poseer:

- Permitir el uso de televigilancia, ya que el sistema tomará fotos cuando se lo requiera, y además mostrará 4 alarmas diferentes en caso de intrusión puesto que son 4 los sensores requeridos que se instalarán.
- Permitir controlar 14 dispositivos eléctricos a distancia mediante el teléfono móvil.
- Permitir simular presencia.
- Permitir controlar el hardware desde una aplicación de telefonía móvil.
- Permitir usar el teléfono móvil como llave electrónica.
- Permitir configurar localmente los usuarios autorizados vía un midlet.
- Permitir configurar remotamente los números autorizados mediante una página web desde Internet.
- Permitir la autenticación del usuario cada vez que se establezca una nueva conexión.

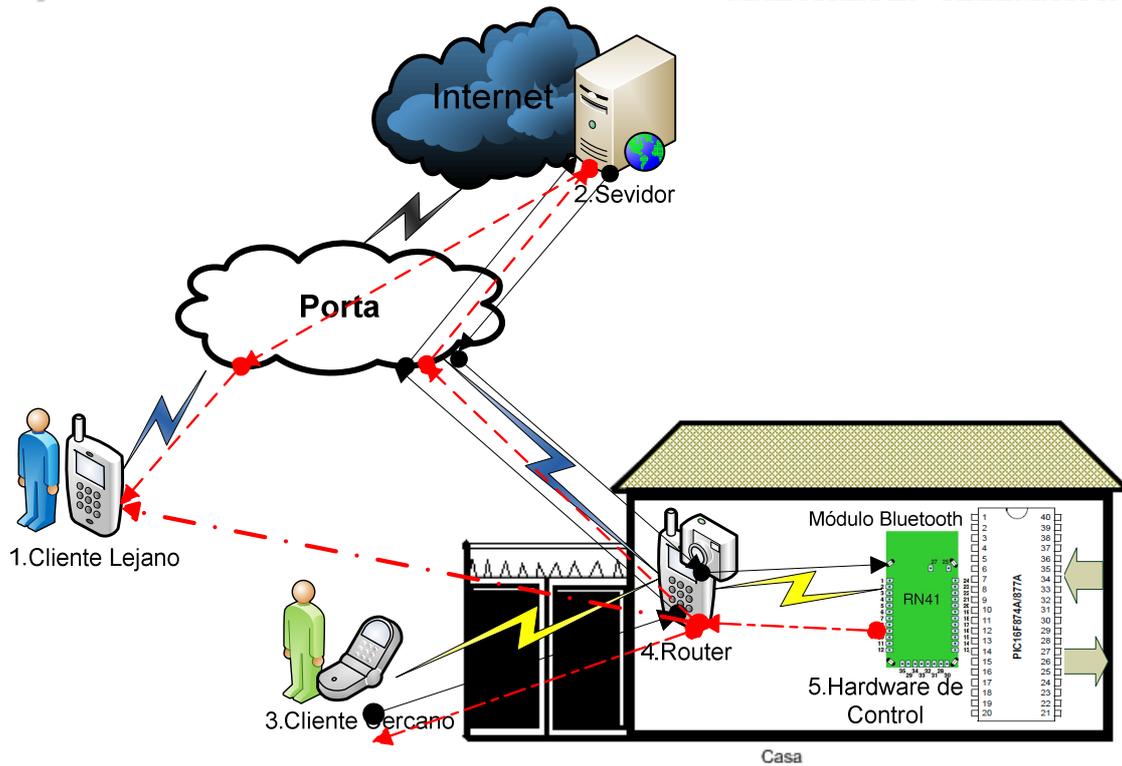


Figura 4-1 Análisis del problema: Interacción cercana.

Leyenda Telecontrol (línea negra):

3-4: El Cliente cercano envía una orden al Router, con la intención de controlar uno de los 14 dispositivos.

4-2: El Router va hasta la base de datos (mobile.sql) en el Servidor, y verifica que el usuario que está intentando acceder al Hardware de Control es un usuario autorizado y por tanto registrado en una tabla de la base de datos.

2-4: Baja un comando con el estado de una bandera, que le permite saber al Router si el usuario estaba autorizado para el uso del sistema o no.

4-5: En caso de que sea usuario autorizado, el Router envía la orden al Hardware de Control para controlar el estado de uno de los 14 dispositivos.

Leyenda Telemedición (línea roja):

5-4: Uno de los cuatro sensores se ha activado, cambiando el estado de uno de los cuatro sensores de 1 a 0, por tanto el Hardware de Control informa de esta alerta al Router mediante un comando, en el cual se especifica cuál de los cuatro sensores ha sido activado (S1, S2, S3, S4). En un diseño más minucioso convendría programar en



éste paso, el encendido de todas las luminarias de la casa para facilitar la captura de imágenes.

4-3: El Router vía Bluetooth, informa al Cliente cercano de la alerta.

4-1: El Router vía un SMS (programado), informa al Cliente lejano del estado de alerta en su domicilio.

4-2: El Router vía un comando GPRS publica en la Web, el estado de los sensores para saber con exactitud cuál sensor fue activado. Además de enviar 5 fotografías tomadas al momento de dispararse la alarma.

2-1: El Cliente lejano abre una página HTML en su móvil, que le permite informarse de lo que está sucediendo en su domicilio, inclusive ver unas cuantas capturas de imágenes (5 fotos).

En la interacción cercana mostrada en la Figura 4-1, para telecontrol (es decir en sentido de ida), observamos que el Cliente cercano es el que inicia el funcionamiento del sistema mediante un comando vía Bluetooth que llega hasta el router, va hasta el servidor a validar usuario y contraseña, y en caso de que la autenticación al sistema sea correcta permite el envío de un comando desde el Router hasta el Hardware de control; y, por lo tanto el accionamiento (o deshabilitación) de uno de los 14 actuadores.

En la interacción cercana para telemedición (es decir en sentido de vuelta), observamos que el accionamiento de un sensor en el hardware es el que inicia el funcionamiento del sistema mediante un comando enviado vía Bluetooth que llega hasta el Router, y es el accionador de los siguientes tres subprocesos:

1. Avisa al Cliente cercano del accionamiento de un sensor en el hardware vía un telemando Bluetooth.
2. Avisa al Cliente lejano del accionamiento de un sensor en el hardware vía un mensaje SMS.
3. Envía fotos desde el Router hasta el Servidor, para capturar imágenes del intruso, o fenómeno físico que disparó la alarma.

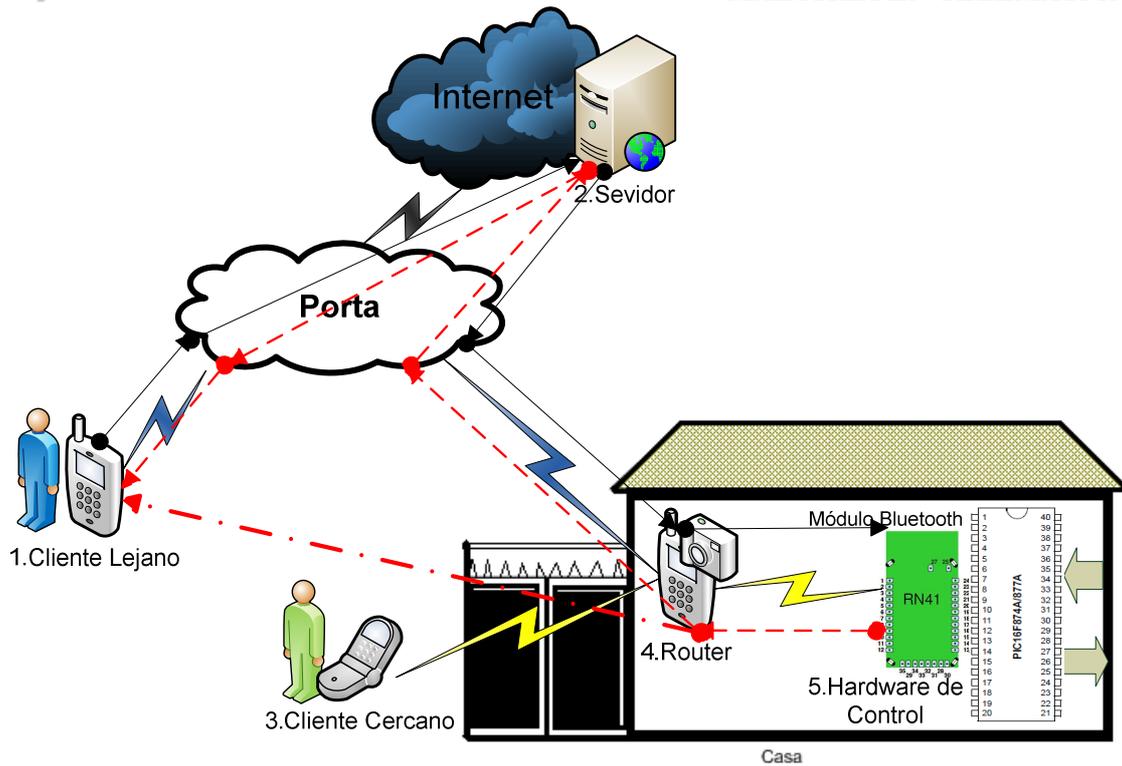


Figura 4-2 Análisis del problema: Interacción lejana.

En la Interacción lejana mostrada en la Figura 4-2 (telecontrol), quien inicia el funcionamiento del sistema, es el Cliente lejano que mediante un telemando vía la red GPRS, llega hasta el Servidor en Internet el mismo que autentica al usuario y luego en caso de que sea usuario registrado envía un telemando al Router indicándole que accione uno de los 14 dispositivos; entonces vía Bluetooth este telemando va desde el Router hasta el Hardware de control.

En la interacción lejana (telemedición), quien inicia el funcionamiento del sistema es la activación de un sensor del Hardware de control, el mismo que con un comando de control llega hasta el Router vía Bluetooth, y en tal caso sucede lo siguiente:

1. El Router, envía una alerta al Cliente lejano vía SMS, para alertarlo de la situación en la que se encuentra el domicilio.
2. El Router, envía 5 fotos hasta el Servidor, capturando las imágenes del intruso.

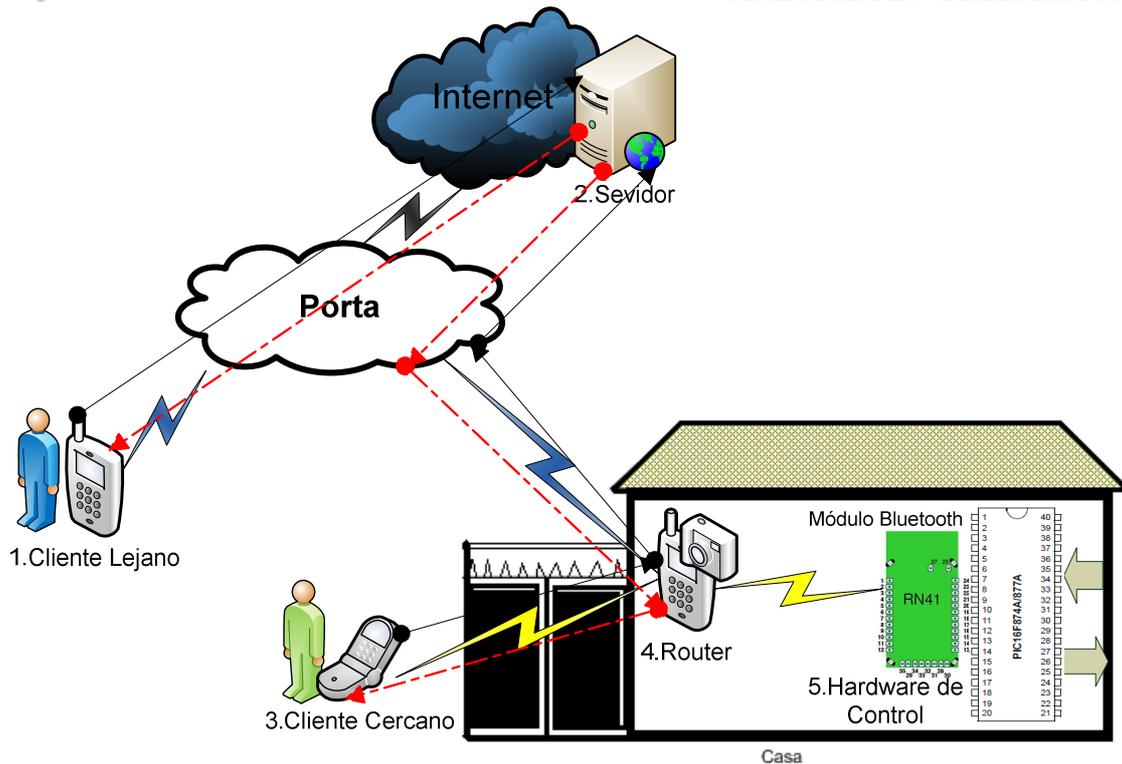


Figura 3-3 Análisis del problema: Registro de usuarios lejano y cercano.

En el Registro de usuarios lejano y cercano mostrado en la Figura 3-3, tenemos que quienes inician el funcionamiento del sistema son ambos móviles (móvil cercano, mediante un midlet, o lejano a través de un navegador y conexión a Internet), primeramente se autentican, y luego crean la nueva cuenta de acceso.

Desde este capítulo en adelante vemos como se va haciendo cada vez más complejo el sistema, y para poder explicarlo fácilmente se ha dividido el sistema en tres escenarios, el primero denominado interacción cercana, el segundo interacción lejana, y el tercero registro de usuarios lejano y cercano. Ahora entendamos que cada escenario se lo ha dividido en dos casos de uso de la siguiente forma:

1. Interacción cercana.

- Caso de uso telecontrol cercano (vía Bluetooth).
- Caso de uso telemedición cercana (vía Bluetooth).

2. Interacción lejana.

- Caso de uso telecontrol lejano (vía GPRS).
- Caso de uso telemedición cercana (vía GPRS).



3. Registro de usuarios lejano y cercano.

- Caso de uso registro lejano (vía GPRS).
- Caso de uso registro cercano (vía Bluetooth y GPRS).

En el siguiente capítulo veremos los casos de uso que nos permitirán establecer con exactitud los requerimientos de cada funcionalidad.



CAPÍTULO 5: DISEÑO DEL SOFTWARE DEL SISTEMA.

5.1 Diseño para la interacción cercana.

5.1.1 Caso de uso telecontrol cercano (vía Bluetooth).

En los casos de uso del sistema vale recalcar que el orden de ejecución de las actividades es de arriba hacia abajo (como es común en esta herramienta de diseño de software).

Caso de uso: Telecontrol cercano

Actores: Cliente cercano, Router, Servidor, Hardware de Control.

Propósito: Activar uno de los 14 dispositivos comandados por el Hardware de control.

Descripción: El Cliente cercano envía un comando hasta el Router, el Router lo redirecciona al Servidor, en el Servidor se valida el usuario y contraseña con el que llega el comando, en caso de ser satisfactoria la validación, retorna un comando desde el Servidor al Router, mismo que es reenviado al Hardware de control para finalmente poder activar o desactivar un dispositivo comandado por el Hardware de control. Todo esto lo apreciamos con más claridad en la Figura 5-1.

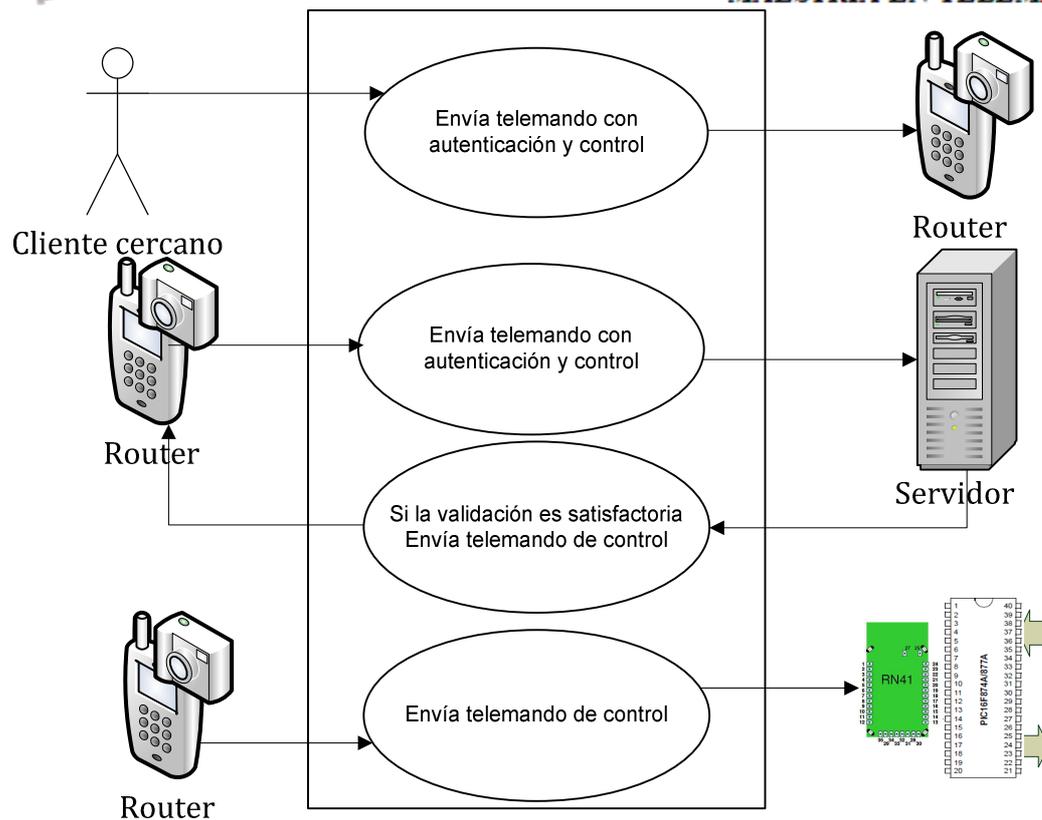


Figura 5-1 Caso de uso Telecontrol cercano.

5.1.2 Caso de uso telemedición cercana (vía Bluetooth).

Caso de uso: Telemedición cercana.

Actores: Cliente cercano, Cliente lejano, Router, Servidor, Hardware de Control.

Propósito: Transmitir cuatro diferentes alertas originadas en el Hardware de control, hasta el Cliente cercano y Cliente lejano; así como el envío de 5 fotografías capturadas por la cámara del Router hasta el Servidor.

Descripción: El Hardware de control envía la alerta al Router, luego el Router difunde la alarma tanto al Cliente cercano, como al Cliente lejano. Adicional a estas dos alarmas, el Router toma 5 fotografías y las envía al Servidor, y allí se almacenan.

Todo esto lo apreciamos con más claridad en la Figura 5-2.

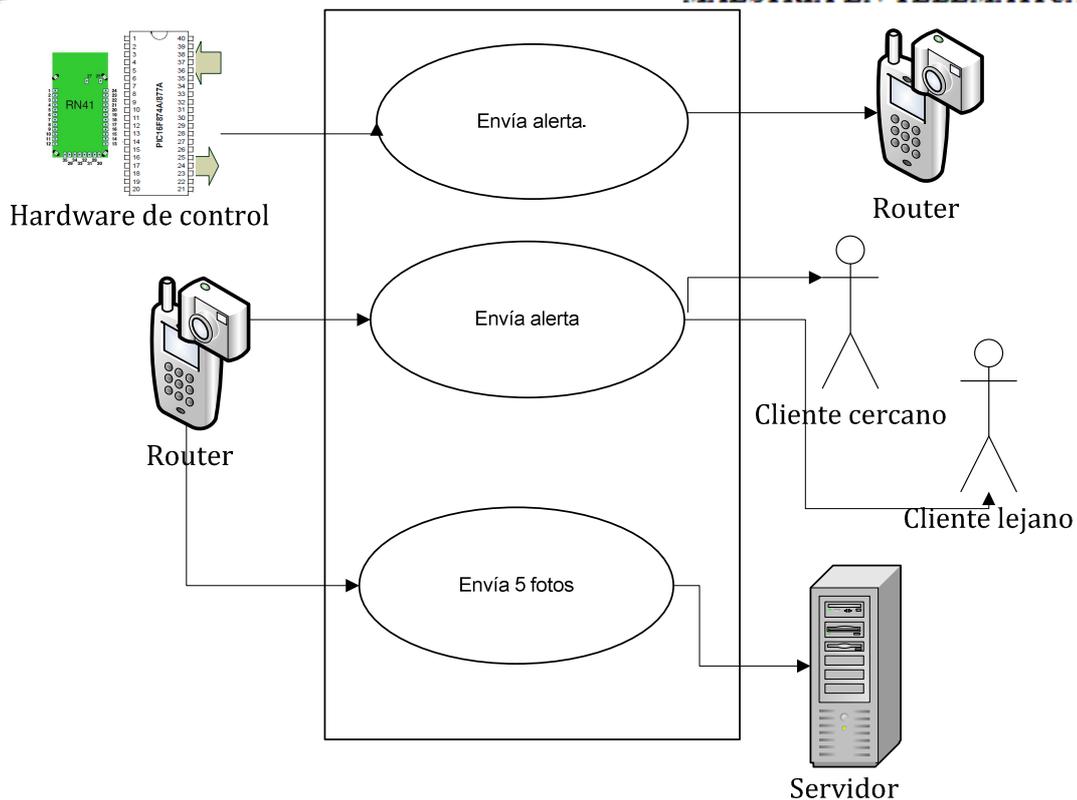


Figura 5-2 Caso de uso Telediccción cercana.

5.2 Diseño para la interacción lejana.

5.2.1 Caso de uso telecontrol lejano (vía GPRS).

Caso de uso: Telecontrol lejano.

Actores: Cliente lejano, Servidor, Router, Hardware de Control.

Propósito: Controlar 14 dispositivos comandados por del Hardware de Control desde un navegador de internet.

Descripción: El cliente lejano envía un comando hasta el servidor, en dicho comando se encuentran usuario, contraseña y el comando de control propiamente dicho. Luego si la autenticación es correcta el comando de control pasa desde el Servidor al Router, y del Router al Hardware de control para cambiar de estado uno de los 14 dispositivos. Todo esto lo apreciamos con más claridad en la Figura 5-3.

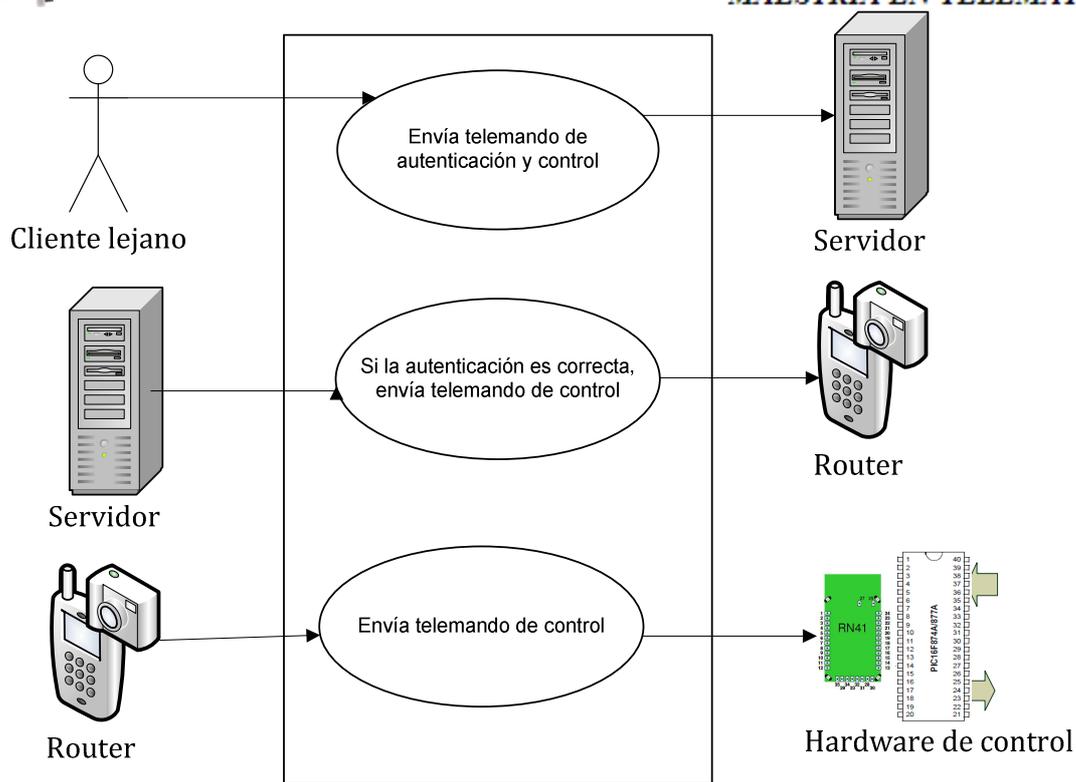


Figura 5-3 Caso de uso Telecontrol lejano.

5.2.2 Caso de uso telemedición lejana (vía GPRS).

Caso de uso: Telemedición lejana.

Actores: Hardware de Control, Router, Cliente lejano, Servidor.

Propósito: Emitir alarmas hasta el Cliente lejano y 5 fotos al Servidor.

Descripción: El Hardware de control, envía una alerta al Router, dicha alerta contiene el nombre del sensor que fue activado (S1, S2, S3, S4). Estando la alerta en el Router es enviada (mediante un SMS) hasta el Cliente lejano, de igual forma desde el Router se envían 5 fotografías hasta el servidor (mediante GPRS). Todo esto lo apreciamos con más claridad en la Figura 5-4.

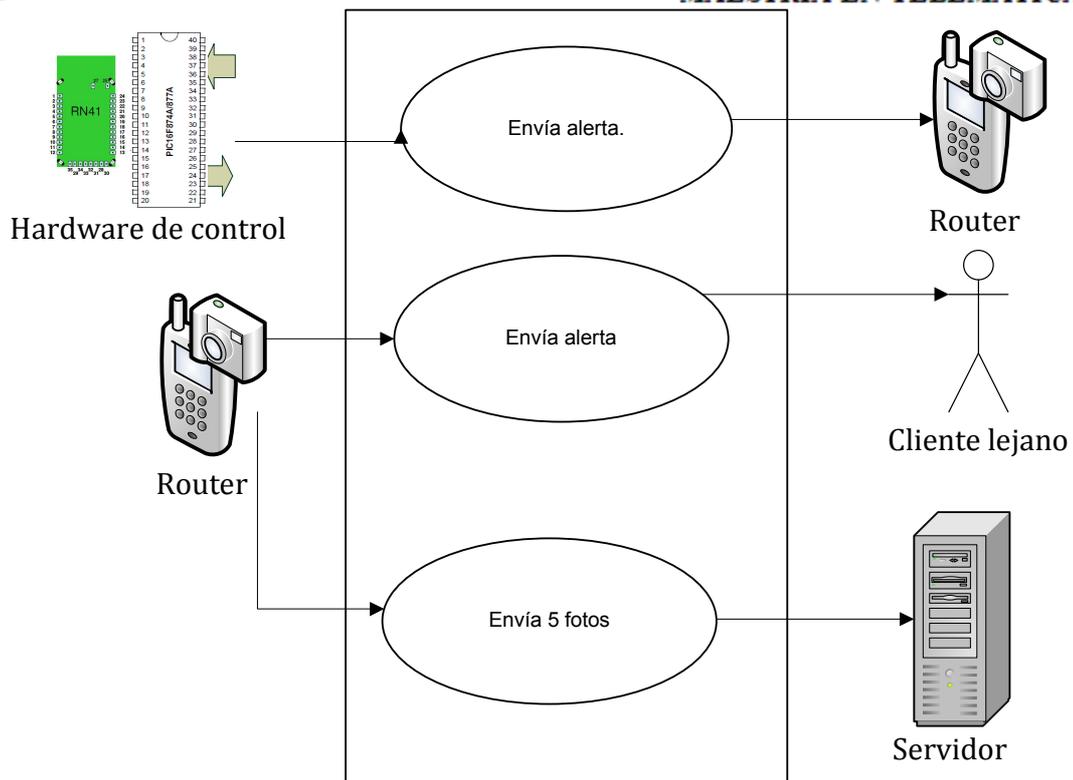


Figura 5-4 Caso de uso Telemedición lejana.

5.3 Diseño para Registro de nuevos usuarios.

4.3.1 Caso de uso registro lejano (vía GPRS).

Caso de uso: Registro lejano.

Actores: Cliente lejano y Servidor.

Propósito: Registrar un nuevo usuario y contraseña en la BD que se encuentra en el Servidor.

Descripción: El Cliente lejano se autentica en el sistema, luego registra el nuevo usuario y contraseña del nuevo usuario autorizado. A continuación el Servidor confirma el registro del nuevo usuario. Todo esto lo apreciamos con más claridad en la Figura 5-5.

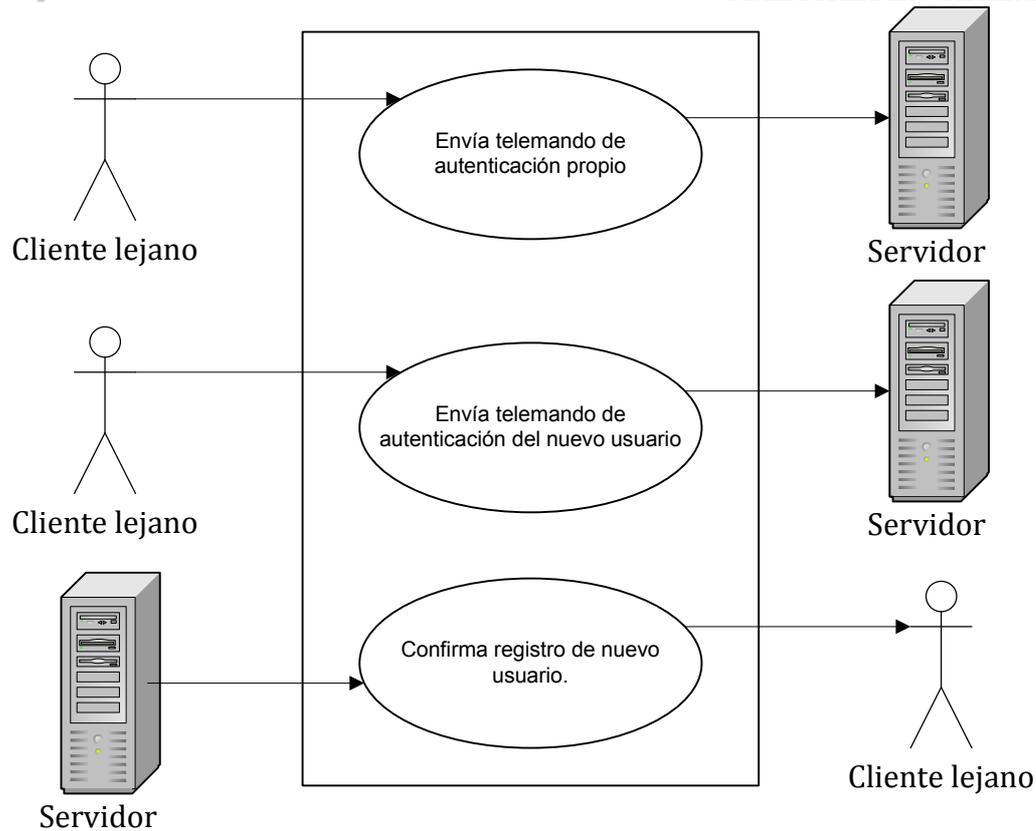


Figura 5-5 Caso de uso Registro lejano.

4.3.2 Caso de uso registro cercano (vía Bluetooth y GPRS).

Caso de uso: Registro cercano.

Actores: Cliente cercano, Router y Servidor.

Propósito: Registrar un nuevo usuario y contraseña en la BD que se encuentra en el Servidor.

Descripción: El Cliente cercano se autentica en el sistema pasando por el Router, luego registra el nuevo usuario y contraseña en la BD del Servidor igualmente pasando por el Router. Finalmente el Servidor le confirma al Cliente cercano que el nuevo usuario ha sido registrado. Todo esto lo apreciamos con más claridad en la Figura 5-6.

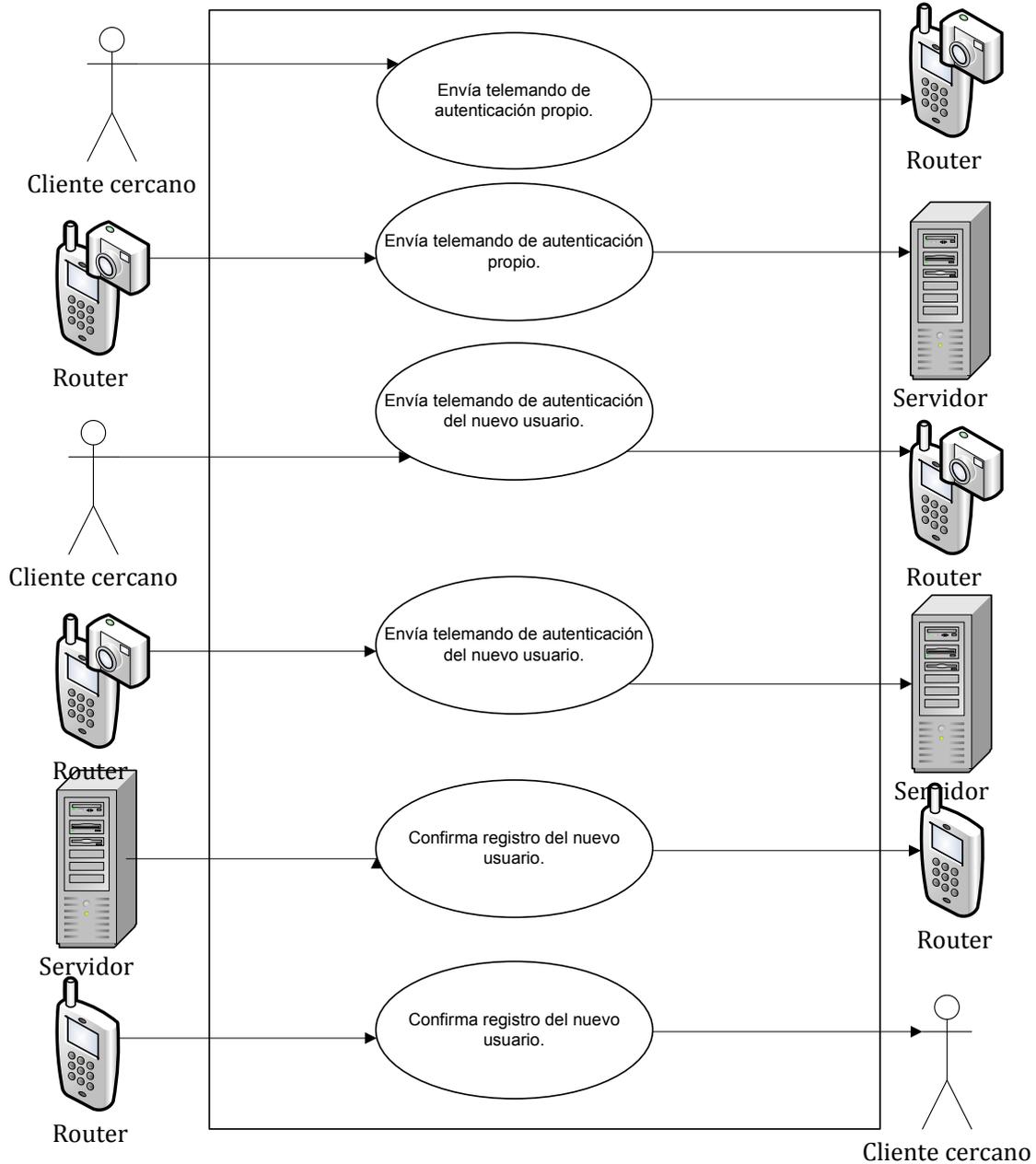


Figura 5-6 Caso de uso Registro lejano.

5.4 Selección del lenguaje y SGBD.

Se ha seleccionado Java por su importancia que cada vez va teniendo en la Web, y este proyecto consta de varias interacciones a través de Internet, así mismo Java es ideal para dispositivos móviles; es común en un teléfono móvil celular encontrar juegos, y varias aplicaciones desarrolladas en Java (J2ME).

Se ha seleccionado MySQL por ser un SGBD, presente en la mayoría de los servidores de hosting, es un software libre, relacional, multihilo y multiusuario; por todas estas



características es que se lo ha elegido, en sí es muy utilizado en aplicaciones web como Drupal y Joomla. En plataformas Linux/Windows, su popularidad como aplicación web está muy ligada a PHP.

5.5 Selección del dominio y hosting.

En los Anexos A-6, A-7 y A-8, observamos información respecto a una página web para verificar que nuestro dominio pueda existir en la web (disponibilidad), la página principal del proveedor de hosting, y la activación de la cuenta por parte del proveedor del servicio de hosting respectivamente.

Una característica particular del servidor de hosting usado, es que es un servidor con Apache, Tomcat y MySQL. Principalmente lo que se usa la aplicación web java, es el Tomcat, ya que permite la ejecución de aplicaciones.

5.6 Configuración de hardware y redes.

En cuanto al hardware, se ha usado móviles que soporten Bluetooth, cámara, y tengan soporte para páginas web. Para ser más precisos, el que en realidad requiere necesariamente el soporte Bluetooth es el Cliente cercano; el que requiere obligatoriamente soporte para leer páginas web es el Cliente lejano, y en cambio el Router lo que requiere es soporte para la cámara, Bluetooth y comunicación vía paquete de datos.

En cuanto a las redes, vale indicar que se usa la red de PORTA, por ser la que mayor y mejor cobertura da por el momento en nuestro país. Con la evolución de las infraestructuras de red de comunicaciones móviles (LTE), poco a poco permitirá una mayor velocidad de transmisión, y por lo tanto cada vez más factible de implementar la transmisión de video y no solo de fotos (imágenes) en éste tipo de sistemas.



CAPÍTULO 6: IMPLEMENTACIÓN DEL SOFTWARE DEL SISTEMA.

6.1 Implementación para el Servidor.

6.1.1 Codificación de la aplicación web en Java.

6.1.1.1 Lectura de imágenes provenientes del Router.

El procedimiento es el siguiente, el Router continuamente está interactuando con el servlet en el Servidor, y de esta manera el Router o bien le entrega una imagen codificada, o bien el comando "nocapturada" (que en código sería la variable `nocapturada`); en realidad, este último comando funciona como una bandera que al activarse lo que hace es parar la toma de fotografías, y al desactivarse permite la recepción de fotografías desde el Router.

Una vez que la imagen llega al servlet en el Servidor, se la decodifica (porque previamente se la codificó en el Router antes de enviarla) a la imagen y luego se la guarda en el disco del Servidor. Entonces en cada interacción se va verificando si la bandera "nocapturada" está activa o no, y dependiendo de eso el Router toma o no una foto para enviarla al servidor.

Algo importante a mencionar en lo referente al envío y recepción de imágenes, es que para transmitir o recibir la imagen, el tratamiento básico que se le da, es una codificación o decodificación con el algoritmo Base64, cuya implementación de este algoritmo en Java es de autoría de Stefan Haustein, Oberhausen, y Rhld; y, es así que con el uso de esta clase, se consiguió enviar imágenes a un servidor en Internet al momento que se active uno de los cuatro sensores del Hardware de control, para obtener algunas fotografías del intruso, o fenómeno físico que inició el disparo de la alarma.

El algoritmo Base 64, consiste en pasar un número en base 10 a base 64, manteniendo una relación de equivalencia entre número y carácter, por ejemplo A=10, B=11, .../=63. El algoritmo Base 64 equivale a hacer la conversión de número a palabras, para lograrlo se basa en el método de divisiones sucesivas hasta llegar a un cociente 0, los restos en orden inverso, expresados como caracteres, nos dan la palabra resultante.



Otro detalle a tomar en cuenta es que el valor de la bandera se almacena en la base de datos del servidor; ya que la base de datos, se la ha implementado con la finalidad de registrar el estado de las variables que controlan los dispositivos.

Algunos detalles de esta codificación son los siguientes.

- `public class ImageServlet extends HttpServlet {` , lo que indica que toda esta clase extiende métodos de la clase `HttpServlet`, que es una clase que contiene métodos con funcionalidades para la web.

- `String s = buf.toString();` , donde `buf`, es un buffer para cadenas de caracteres entrante, exactamente en la variable `s`, es lo que o bien se receipta caracteres correspondientes a una imagen, o bien se recibe la palabra `nocapturada`, que impide la toma de fotografías.

- `if(s!="nocapturada"){`
`objNuevoPng = new NuevoPng(arrIm, s);`
`objNuevoPng.start();`

`}`, lo que nos indica que si lo que recibe en el buffer del servlet es el comando `nocapturada`, entonces impide la toma de fotografías; y en caso de que la cadena del buffer de entrada al servlet sea diferente de `nocapturada`, entonces ejecuta `NuevoPng`, que lo que hace es decodificar y guardar la cadena entrante correspondiente a la fotografía.

- `writer.write(r_salida);` , nos permite escribir al Router, para cambiar el estado de su bandera para la toma y envío de fotografías.

6.1.1.2 Lectura y presentación de la imagen guardada en el Servidor.

Anteriormente ya analizamos la llegada de la imagen al servidor, lo que vemos a continuación en la Figura 6-1, es la presentación en pantalla de dicha imagen.



Figura 6-1 Imagen capturada desde el Router.

Los botones Capturar y Parar, son los que nos permiten cambiar el estado del Router para predisponerlo o no a la toma y envío de imágenes al Servidor.

Algo más acerca de la bandera del Router, es que se almacena su estado en la base de datos, entonces cada cambio de estado en un dispositivo se reflejará en la Base de Datos MySQL, que viene disponible en el Servidor Web.

Igualmente, en lo referente a la publicación de imágenes guardadas en disco, tenemos que la clase para esta funcionalidad adopta métodos de la clase HttpServlet.



6.1.1.3 Tablas de la Base de Datos.

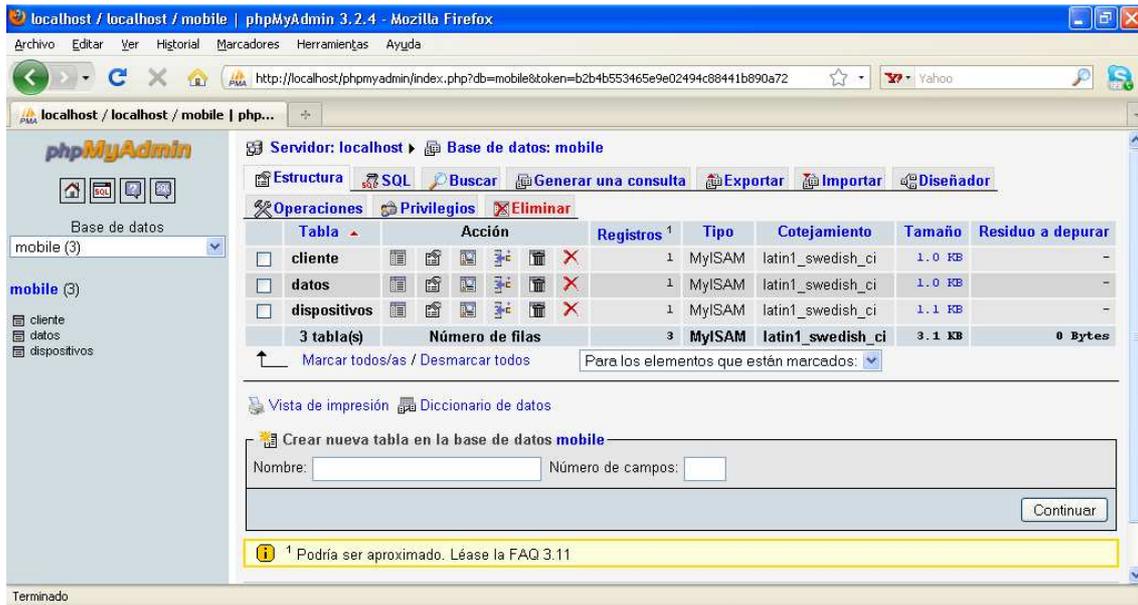


Figura 6-2 Base de datos del sistema.

En la Figura 6-2 observamos, la existencia de tres tablas de la base de datos, necesarias para almacenar todos los estados y configuraciones del sistema. En la parte izquierda observamos que el nombre de la base de datos es "mobile", la cual contiene tres tablas: cliente, datos y dispositivos.

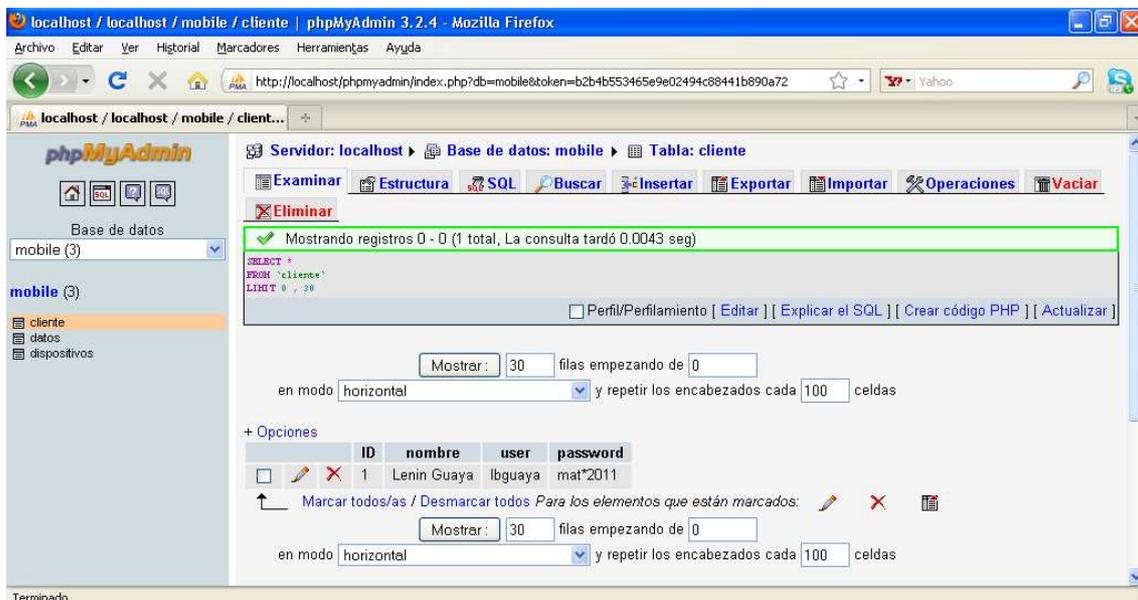


Figura 6-3 Tabla cliente de la Base de Datos.



En la Figura 6-3 observamos que por el momento existe solamente un usuario autorizado a ingresar al sistema, cuyo usuario es "lbguaya" y el password es "mat*2011". Aquí se aprecia el nombre de cada campo, así como el contenido del primer registro.

Tanto para el logeo del cliente lejano como cercano la consulta se la hace hasta esta tabla de la base de datos.

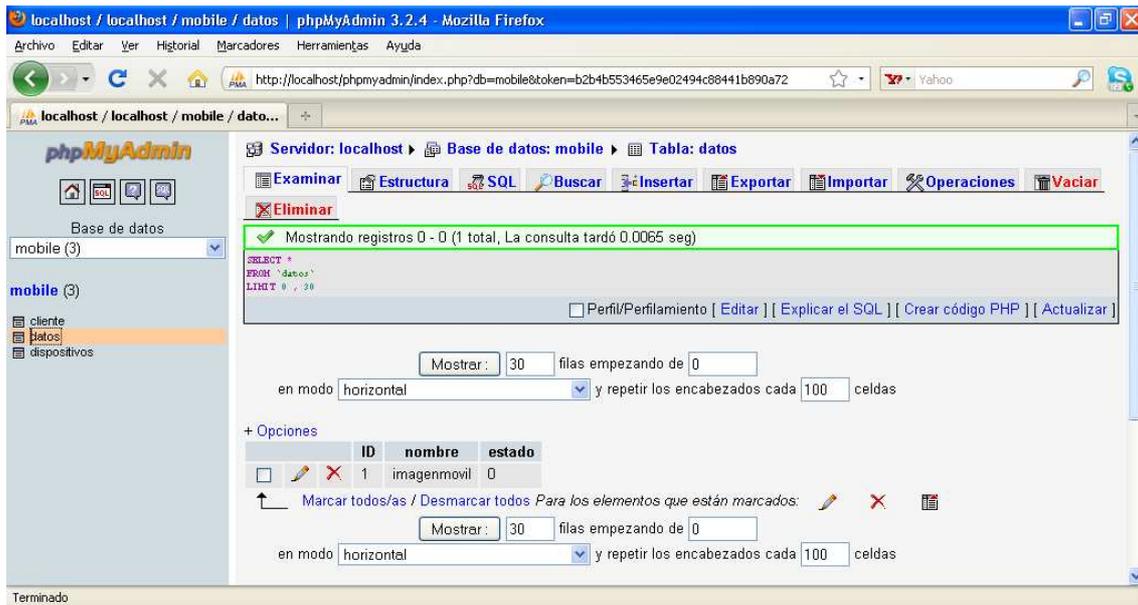


Figura 6-4 Tabla Datos de la Base de Datos.

En la Figura 6-4 tenemos la tabla datos, la misma que nos permite guardar el estado de captura o no de una imagen. Si el estado está en 1 capturará fotos, si está en cero no lo hará.

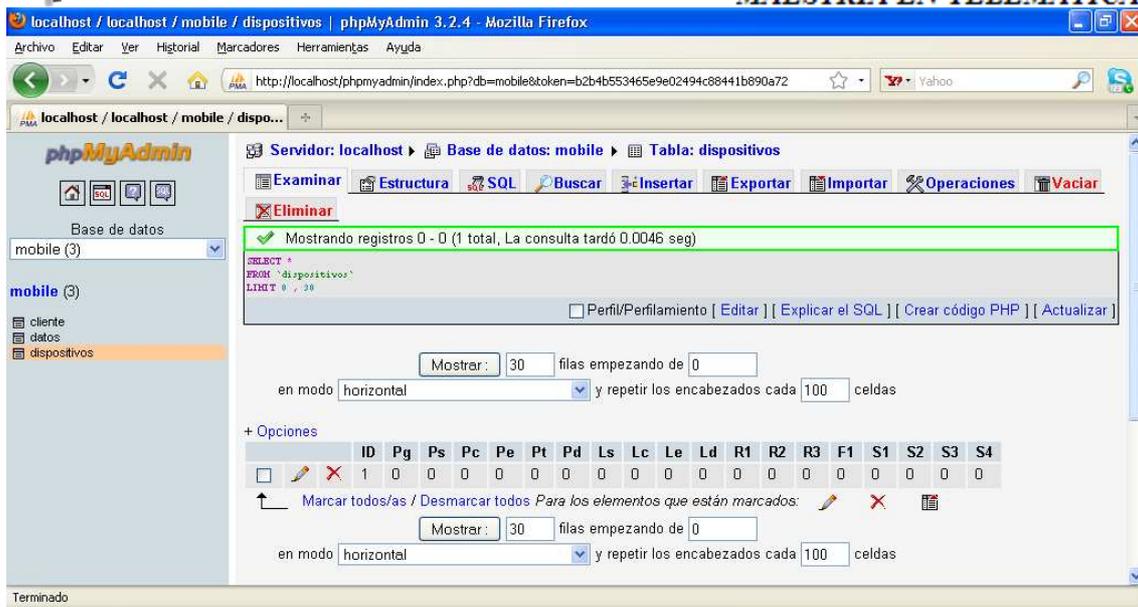


Figura 6-5 Tabla dispositivos de la Base de Datos.

En la Figura 6-5 tenemos la tabla dispositivos, en la cual observamos que los diez y ocho dispositivos están deshabilitados.

6.2 Implementación para el Celular Router.

6.2.1 Codificación del midlet para el Router.

6.2.1.1 Envío de la imagen al servidor desde el Router.

Algunos detalles de codificación son:

- String entrada="1"; , es una petición de fotografía desde el servlet (en el Servidor) para el Router. Cuando el Servidor envía la petición entrada=1, el Router envía una imagen al Servidor. Mientras tanto, cuando entrada=0 (petición desde el Servidor), no toma la foto y en tal caso el servlet en el Servidor envía la cadena nocapturada.
- public class CameraMID extends MIDlet implements CommandListener, Runnable {, lo cual nos indica que CameraMID, es un midlet (archivo a ejecutar), e implementa funcionalidades para el enlistado y uso de threads, el hilo de ejecución se lo usa para mientras recibo (desde el Servidor) una cadena que me indica el estado de la bandera del Router, poder capturar y enviar la imagen al Servidor.



- ("http://localhost:8084/Servidor/ImageServlet");, esta URL, hace referencia a mi propio equipo de simulación, al cual accediendo por el puerto 8084, me permite la ejecución del servlet denominado ImageServlet.

Ya que la ejecución la estamos haciendo en el simulador, al no estar capturando video por la cámara del celular Router, se genera automáticamente una secuencia de video, a partir de la cual se simula la toma de fotografías.

En la gráfica de la Figura 6-6, observamos las propiedades del Router, principalmente centrémonos en el CLDC y MIDP, así como en el Wireless Toolkit 2.5.2 for CLDC, el mismo que nos permite simular el dispositivo físico.

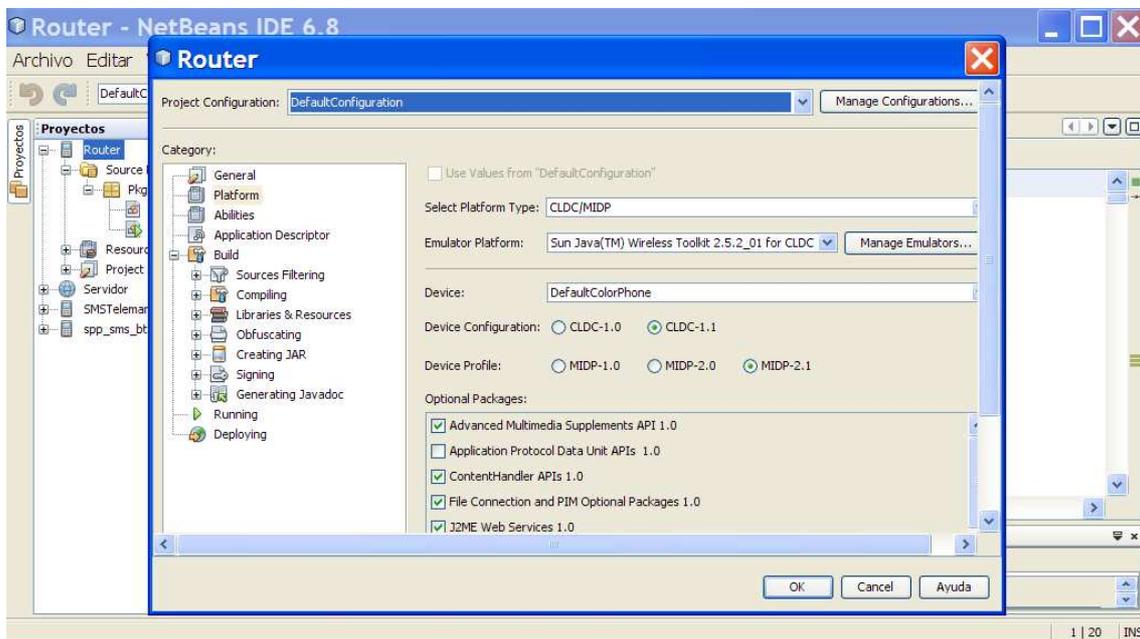


Figura 6-6 Propiedades del Router.

A continuación en la Figura 6-7, vemos la simulación de captura de video y envío hacia el Servidor en Internet, en donde a partir del video generado es lo que se va tomando fotografías y enviando al Servidor.



Figura 6-7 Simulando la captura de imágenes desde el Router.

6.2.1.2 Envío de un SMS de alerta al cliente lejano.

EnvioSMS.java, es la clase que implementa la interfáz runnable que permite ejecutar hilos, específicamente, el método run() que a su vez ejecutará el método de envío de mensajes SMS, así lo observamos claramente en el código fuente de la Figura 6-8.

```
package SMS;  
import javax.microedition.io.*;  
import javax.wireless.messaging.*;  
import java.io.IOException;
```



/*Clase que implementa propiedades para manipular hilos, recordemos, que la implementación es una solución a la herencia múltiple*/

```
public class envioSMS implements Runnable {
    private static envioSMS inst = new envioSMS();

    public static envioSMS getInstance() {
        return inst;
    }
    private String receptor = null;
    private String puerto = null;
    private String mensaje = null;
    private boolean mSending = false;

    /*
     * Método para el envío del mensaje. Recibe como parámetros el número de
     * teléfono del receptor, el número de puerto de envío de mensajes SMS,
     * y el contenido del mensaje.
     * Finalmente creamos un hilo y lo inicializamos.
     * @param receptor
     * @param puerto
     * @param mensaje
     */
    public void sendMsg(String receptor, String puerto, String mensaje) {
        if (mSending) return;//verificación de hilo inicializado
        this.receptor = receptor;
        this.puerto = puerto;
        this.mensaje = mensaje;
        Thread th = new Thread(this);
        th.start();
    }

    public boolean isSending() {
        return mSending;
    }
}
```



}

/*

* Método que se ejecuta cuando se inicializa un hilo

*/

```
public void run() {
```

```
    mSending = true;
```

```
    try {
```

```
        sendSMS();
```

```
    } catch (Exception ex) {
```

```
        ex.printStackTrace();
```

```
        System.out.println("error al enviar el mensaje");
```

```
    }
```

```
    mSending = false;
```

```
}
```

/*

* Método para el envío del mensaje

*/

```
private void sendSMS() {
```

```
    String address = "sms://" + receptor + ":" + puerto; //URL
```

```
    MessageConnection conn = null; //variable de conexión
```

```
    try {
```

```
        /* Abrimos la conexión para el envío del SMS */
```

```
        conn = (MessageConnection) Connector.open(address);
```

```
        TextMessage txtmessage = (TextMessage)
```

```
conn.newMessage(MessageConnection.TEXT_MESSAGE);
```

```
        txtmessage.setAddress(address);
```

```
        txtmessage.setPayloadText(mensaje);
```

```
        conn.send(txtmessage);
```

```
    } catch (Throwable t) {
```

```
        System.out.println("Send caught: ");
```

```
        t.printStackTrace();
```

```
}
```



```
if (conn != null) { //Si existe una conexión abierta
    try {
        conn.close();//cierro la conexión
        System.out.println("mensaje enviado con éxito.....");
    } catch (IOException ioe) { //capturador de excepción
        ioe.printStackTrace();//muestra la excepción
        System.out.println("error al enviar el mensaje");
    }
}
}
```

Figura 6-8 Código fuente en J2ME de la clase que envía alertas SMS al Cliente lejano.

Entonces, a partir de la clase analizada es lo que podemos enviar mensajes SMS al Cliente lejano, para que el cliente lejano a partir de esa alerta se conecte a Internet desde un celular con soporte de páginas HTML o desde una portátil con navegador, y de esta forma pueda ver las fotografías que se han capturado.

6.2.1.3 Telemandos de interacción entre el Router y el Servidor.

Todos los telemandos enviados entre el Router y el Servidor se ven reflejados en la base de datos del Servidor, es decir, el estado de cada uno de los 18 dispositivos se ve reflejado aquí en la base de datos de MySQL que se encuentra en el Servidor.

Un poco más complicado fue el envío de la fotografía del Router al Servidor, así que el envío de comandos nada más, es más sencillo y se lo realiza de manera similar a cuando enviábamos los caracteres que conforman la fotografía codificada.

6.2.1.4 Telemandos de interacción entre el Router y el Hardware.

Las interacciones con el hardware se las hace a través de SPPCliente.java, y MidletClienServ.java, donde éste último es el midlet que se ejecuta.

SPPCliente.java, es la clase que se muestra en la Figura 6-9; código fuente que crea una lista con las opciones de búsqueda, enviar, y retroceder para adjuntarla desde el formulario principal (MidletClienServ.java).

A continuación se muestra la clase SPPBluetooth.java



```
package SPPBluetooth;
/*Librería para construir la interfaz de la aplicación*/
import javax.microedition.lcdui.*;
/*Librería que contiene las clases y métodos para trabajar con
bluetooth*/
import javax.bluetooth.*;
/*La clase principal extiende funcionalidades de List*/
public class SPPCliente extends List{
    public SPPCliente(){
        super("Cliente SPP",List.EXCLUSIVE);
        addCommand(new Command("Busqueda",Command.SCREEN, 1));
        addCommand(new Command("Enviar",Command.SCREEN, 1));
        addCommand(new Command("Atras",Command.EXIT, 1));
        //this.setCommandListener(SPPClienteMIDlet.SPPc);
        this.setCommandListener(MidletClienServ.SPPs);
    }
    //Este método se encarga de limpiar la pantalla y de mostrar un mensaje
    public void escribirMensaje(String str){
        for(int i=0;i<this.size();i++) delete(i);
        append(str,null);
    }
    //Este método muestra los "friendly names" de los dispositivos remotos
    public void mostrarDispositivos(){
        for(int i=0;i<this.size();i++)
            delete(i);
        if(MidletClienServ.dispositivos_encontrados.size(>0)){
            for(int i=0;i<MidletClienServ.dispositivos_encontrados.size();i++){
                try{
                    RemoteDevice                dispositivoRemoto                =
                    (RemoteDevice)MidletClienServ.dispositivos_encontrados.elementAt(i);
                    append(dispositivoRemoto.getFriendlyName(false),null);
                }
            }
        }
    }
}
```



```
append((String) MidletClienServ.dispositivos_encontrados.elementAt(i),
null);

    }catch(Exception e){
        System.out.println("Se ha producido una excepcion");
    }
    //RemoteDevice                dispositivoRemoto                =
(RemoteDevice)MidletClienServ.dispositivos_encontrados.elementAt(i);
    //append("LBGUAYA-PC",null);
    }
}
else
    append("Pulse Búsqueda",null);
}

}
```

Figura 6-9 Clase para interacción entre el Router y el Hardware de Control

A continuación, en la Figura 6-10, se presenta el midlet que se comunica con el hardware, claro está que complementándose con la clase anterior de la Figura 6-9. Esto lo podemos entender claramente al leer la documentación implícita tanto de la clase como del midlet.

Lo que a continuación se muestra es el MidletClienServ para lo que es el archivo de ejecución que nos permitirá la interacción entre el Router y el Hardware.

```
/*
* En esta clase se construye la interfaz de usuario para
* la aplicación cliente. Se crea una conexión con
* características cliente y servidor, lo cual permite enviar
* y recibir comandos. Desde aquí se cuenta con un menú de
* opciones que nos permite manejar la aplicación pudiendo
* realizar una búsqueda de dispositivos y enviar el telemando.
*/
```



```
package SPPBluetooth;
```

```
import java.util.Vector;
```

```
import javax.bluetooth.*;
```

```
import javax.microedition.io.Connector;
```

```
import javax.microedition.io.StreamConnection;
```

```
import javax.microedition.io.StreamConnectionNotifier;
```

```
import javax.microedition.lcdui.*;
```

```
import javax.microedition.midlet.*;
```

```
import java.io.*;
```

```
/**
```

```
 * MAT 2010.
```

```
 * @author Lenin Guaya Delgado
```

```
 */
```

```
public class MidletClienServ extends MIDlet implements Runnable, CommandListener,  
ItemCommandListener {
```

```
    private boolean listeningServ = true;
```

```
    private Display pantalla = null;
```

```
    public Form frmCliente;
```

```
    private static final Command salir = new Command("Salir", Command.EXIT, 1);
```

```
    private static final Command Go = new Command("Go", Command.OK, 1);
```

```
    private static final Command enviar = new Command("Enviar", Command.OK, 1);
```

```
    public TextField txtMsjEnviado;
```

```
    private TextField txtMsgRecivido;
```

```
    private LocalDevice local_device;
```

```
    private Thread t;
```

```
    //Servidor
```

```
    public LocalDevice dispositivoLocal;
```

```
    public DiscoveryAgent da;
```

```
    public boolean fin = false;
```



```
public StreamConnectionNotifier servidor;
//private SPPServidor s =null;
public static MidletClienServ SPPs = null;
//Cliente
public static Vector dispositivos_encontrados = new Vector();
public static Vector servicios_encontrados = new Vector();
public static int dispositivo_seleccionado = -1;
private SPPCliente cliente = null;
public DiscoveryAgent da2;
public LocalDevice dispositivoLocal2;

StreamConnection con;
DataOutputStream out;
DataInputStream in;
/*
* Desde este constructor inicializamos variables necesarias
* para colocar los valores correspondientes a las variables
* de la clase.
*/
public MidletClienServ() {
    SPPs = this;
    cliente = new SPPCliente();
    pantalla = Display.getDisplay(this);
    frmCliente = new Form("MSJS BLUETOOTH");
    txtMsjEnviado = new TextField("rsguayax@gmail.com", "", 500,
    TextField.ANY);
    txtMsjEnviado.setItemCommandListener(this);
    txtMsgRecivido = new TextField("Respuesta:", "", 500, TextField.ANY);
    frmCliente.addCommand(salir);
    frmCliente.addCommand(Go);
    frmCliente.addCommand(enviar);
    frmCliente.append(txtMsjEnviado);
    frmCliente.append(txtMsgRecivido);
}
```



```
frmCliente.setCommandListener(this);
pantalla.setCurrent(frmCliente);
inicializar();
}
/*
* Se inicializan objetos que nos permitirán trabajar
* con la manipulación del dispositivo local como del
* dispositivo remoto. Además creamos un hilo y lo
* inicializamos para crear la conexión servidora.
*/
public void inicializar() {
    try {
        dispositivoLocal = LocalDevice.getLocalDevice();
        dispositivoLocal.setDiscoverable(DiscoveryAgent.GIAC);
        dispositivoLocal2 = LocalDevice.getLocalDevice();
        dispositivoLocal2.setDiscoverable(DiscoveryAgent.GIAC);
        //Lanzamos un hilo servidor (solo aceptara un cliente)
        Thread hilo = new Thread(this);
        hilo.start();
    } catch (BluetoothStateException be) {
        System.out.println("Se ha producido un error al inicializar el hilo servidor");
    }
}
/*
* Este método ejecuta la conexión servidora al inicializar el
* hilo.
*/
public void run() {
    listeningServ = true;
    //Le damos un nombre a nuestra aplicación
    String nombre = "Proyecto SPP";
    //Definimos un UUID único para este servicio. Elegimos uno a nuestro gusto.
    UUID uuid = new UUID(0xABCD);
```



```
System.out.println(uuid.toString());

servidor = null;//Similar a un socket servidor
StreamConnection sc = null;//Similar a un socket cliente
//Intentamos crear una conexion, usando SPP
try {
    servidor = (StreamConnectionNotifier) Connector.open("btspp://localhost:" +
uuid.toString()
    + ";name=" + nombre);
    //Obtenemos el service record
    ServiceRecord rec = dispositivoLocal.getRecord(servidor);
    //Rellenamos el BluetoothProfileDescriptionList usando el SerialPort version 1
    DataElement e1 = new DataElement(DataElement.DATSEQ);
    DataElement e2 = new DataElement(DataElement.DATSEQ);
    e2.addElement(new DataElement(DataElement.UUID, new
UUID(0x1101));//agregamos el puerto serie
    e2.addElement(new DataElement(DataElement.INT_8, 1));//version 1
    e1.addElement(e2);
    //agregamos al service record el BluetoothProfileDescriptionList
    rec.setAttributeValue(0x0009, e1);
} catch (Exception e) {
    System.out.println("Se ha producido un error al lanzar el hilo servidor");
    mostrarAlarma(e, this.frmCliente, 0);
    return;
}

while (!fin) {
    try {
        sc = servidor.acceptAndOpen();//Se espera a que leguen datos
        //en el buffer de entrada.
        DataInputStream in = sc.openDataInputStream();
        DataOutputStream out=sc.openDataOutputStream();/**/
```



```
while(true){
    byte[] b =new byte[]{in.readByte()};
    String cadena = new String(b);
    txtMsgRecivido.setString(cadena);
    in.close();
    out.close();
    sc.close();
}
} catch (Exception e) {
    mostrarAlarma(e, this.frmCliente, 0);
    return;
}
}

}

public void startApp() {
    pantalla = Display.getDisplay(this);

}

public void pauseApp() {
}

public void destroyApp(boolean unconditional) {
}

//Este método se encarga de dar un aviso de alarma cuando se produce una excepción
public void mostrarAlarma(Exception e, Screen s, int tipo) {
    Alert alerta = null;
    if (tipo == 0) {
        alerta = new Alert("Excepcion", "Se ha producido la excepción " +
e.getClass().getName(), null,
```



```
AlertType.ERROR);
    } else if (tipo == 1) {
        alerta = new Alert("Error", "No ha seleccionado un dispositivo ", null,
AlertType.ERROR);
    } else if (tipo == 2) {
        alerta = new Alert("Información", "El mensaje ha sido enviado ", null,
AlertType.INFO);
    }
    alerta.setTimeout(Alert.FOREVER);
    pantalla.setCurrent(alerta, s);
}

```

/*

* Este método se ejecuta cada vez que seleccionamos una
* opción del menú que se creó en la clase SPPCliente.
* Con la opción de búsqueda se realiza una búsqueda de dispositivos
* que estén a nuestro alcance. Con opción de enviar
* se hace una búsqueda de los servicios ofrecidos por el
* dispositivo seleccionado.
*/

```
public void commandAction(Command co, Displayable d) {
    //pantalla.setCurrent(frmCliente);
    if (co.getLabel().equals("Busqueda")) {
        //Limpiamos la lista

        dispositivos_encontrados.removeAllElements();
        servicios_encontrados.removeAllElements();
        try {
            //dispositivoLocal2 = LocalDevice.getLocalDevice();
            //dispositivoLocal2.setDiscoverable(DiscoveryAgent.GIAC);
            da2 = dispositivoLocal2.getDiscoveryAgent();
            da2.startInquiry(DiscoveryAgent.GIAC, new Listener());
            cliente.escribirMensaje("Por favor espere...");
        } catch (BluetoothStateException be) {

```



```
mostrarAlarma(be, cliente, 0);
}
} else if (co.getLabel().equals("Enviar")) { //else if(d==cliente &&
co.getLabel().equals("Enviar")){
    dispositivo_seleccionado = cliente.getSelectedIndex();
    //Nos aseguramos de que el usuario seleccione un dispositivo
    if (dispositivo_seleccionado == -1 || dispositivo_seleccionado >=
dispositivos_encontrados.size()) {
        mostrarAlarma(null, cliente, 1);
        return;
    }

    servicios_encontrados.removeAllElements();
    //Buscamos el servicio de puerto serie en el dispositivo seleccionado
    RemoteDevice dispositivo_remoto = (RemoteDevice)
dispositivos_encontrados.elementAt(dispositivo_seleccionado);
    try {
        //Buscamos en el puerto serie 0x1101
        da2.searchServices(null, new UUID[]{new UUID(0x1101)},
dispositivo_remoto, new Listener());
    } catch (BluetoothStateException be) {
        mostrarAlarma(be, cliente, 0);
    }

    pantalla.setCurrent(frmCliente);
} else if (co == enviar) { //else if(d==cliente && co.getLabel().equals("Enviar")){
    dispositivo_seleccionado = cliente.getSelectedIndex();
    //Nos aseguramos de que el usuario seleccione un dispositivo
    if (dispositivo_seleccionado == -1 || dispositivo_seleccionado >=
dispositivos_encontrados.size()) {
        mostrarAlarma(null, cliente, 1);
        return;
    }
}
```



```
servicios_encontrados.removeAllElements();
//Buscamos el servicio de puerto serie en el dispositivo seleccionado
RemoteDevice dispositivo_remoto = (RemoteDevice)
dispositivos_encontrados.elementAt(dispositivo_seleccionado);
try {
    //Buscamos en el puerto serie 0x1101
    da2.searchServices(null, new UUID[]{new UUID(0x1101)},
dispositivo_remoto, new Listener());
    } catch (BluetoothStateException be) {
        mostrarAlarma(be, cliente, 0);
    }

    pantalla.setCurrent(frmCliente);
} else if (d == cliente && co.getLabel().equals("Atras")) {
    pantalla.setCurrent(frmCliente);
} else if (co == Go) {
    cliente = new SPPCliente();
    //msg = new Mensaje();
    //Mostramos la lista de dispositivos(vacia al principio)
    cliente.mostrarDispositivos();
    pantalla.setCurrent(cliente);
} else if (co == salir) {
    destroyApp(true);
    notifyDestroyed();
}
}

public void reenviarMensaje() {
    dispositivo_seleccionado = cliente.getSelectedIndex();
    //Nos aseguramos de que el usuario seleccione un dispositivo
    if (dispositivo_seleccionado == -1 || dispositivo_seleccionado >=
dispositivos_encontrados.size()) {
```



```
mostrarAlarma(null, cliente, 1);
return;
}

servicios_encontrados.removeAllElements();
//Buscamos el servicio de puerto serie en el dispositivo seleccionado
RemoteDevice dispositivo_remoto = (RemoteDevice)
dispositivos_encontrados.elementAt(dispositivo_seleccionado);
try {
    //Buscamos en el puerto serie 0x1101
    da2.searchServices(null, new UUID[]{new UUID(0x1101)},
dispositivo_remoto, new Listener());
} catch (BluetoothStateException be) {
    mostrarAlarma(be, cliente, 0);
}

pantalla.setCurrent(frmCliente);
}
//Este método se va a encargar de enviar un mensaje al primer ServiceRecord usando
el
//Serial Port Profile
/*
* Desde éste método creamos un registro del servicio
* del dispositivo seleccionado con los cual formamos la
* URL hacia el dispositivo remoto, y a ésta URL, le
* indicamos que la conexión va a ser autenticada.
*Con lo cual luego creamos una conexión para el envío de comandos.
*/
public void enviarMensaje(String msg) {
    System.out.println(servicios_encontrados.size());
    //for (int i = 0; i < servicios_encontrados.size(); i++) {
        //System.out.println(servicios_encontrados.elementAt(i));
    //}
```



```
ServiceRecord sr = (ServiceRecord) servicios_encontrados.elementAt(0);
```

```
String URL =  
sr.getConnectionURL(ServiceRecord.AUTHENTICATE_NOENCRYPT, false);  
try {  
    //Obtenemos la conexión y el stream de este servicio  
    con = (StreamConnection) Connector.open(URL);  
    out = con.openDataOutputStream();  
    in = con.openDataInputStream();  
  
    //Escribimos datos en el stream  
    out.writeUTF(msg);  
    out.flush();  
    txtMsgRecivido.setString(in.readUTF());  
  
} catch (Exception e) {  
    mostrarAlarma(e, cliente, 0);  
}  
}  
  
public void commandAction(Command c, Item item) {  
  
}  
  
//Implementamos el DiscoveryListener  
/*  
 * Esta clase implementa la interfaz Discovery Listener la  
 * cual contiene los métodos para realizar búsquedas de dispositivos  
 * y servicios.  
 */  
public class Listener implements DiscoveryListener {  
    //Implementamos los métodos del interfaz DiscoveryListener  
  
    /*
```



* Este método permite agregar un nuevo dispositivo a la

* lista actual de dispositivos encontrados.

*/

```
public void deviceDiscovered(RemoteDevice dispositivoRemoto, DeviceClass
clase) {
    System.out.println("Se ha encontrado un dspositivo Bluetooth");
    dispositivos_encontrados.addElement(dispositivoRemoto);
}
```

/*

* Este método se ejecuta cuando se ha acabado de realizar la

* búsqueda de dispositivos y nos muestra en la pantalla la lista

* de todos los dispositivos encontrados.

*

*/

```
public void inquiryCompleted(int completado) {
    System.out.println("Se ha completado la busqueda de dispositivos");
    if (dispositivos_encontrados.size() == 0) {
        Alert alerta = new Alert("Problema", "No se ha encontrado dispositivos", null,
AlertType.INFO);
        alerta.setTimeout(3000);
        cliente.escribirMensaje("Presione descubrir dispositivos");
        pantalla.setCurrent(alerta, cliente);
    } else {
        cliente.mostrarDispositivos();
        pantalla.setCurrent(cliente);
    }
}
```

/*

* Este método realiza una búsqueda de servicios en los dispo-

* sitios encontrados.

*/

```
public void servicesDiscovered(int transID, ServiceRecord[] servRecord) {
    System.out.println("Se ha encontrado un servicio remoto");
```



```
for (int i = 0; i < servRecord.length; i++) {  
    ServiceRecord record = servRecord[0];  
    servicios_encontrados.addElement(servRecord[0]);  
    //System.out.println(record.toString());  
    System.out.println(servRecord.length);  
}  
}  
/*  
* Este método se ejecuta cuando se ha terminado de realizar la  
* búsqueda de servicios en los dispositivos encontrados.  
* Luego se envía la cadea del comando al dispositivo remoto  
* seleccionado.  
*/  
public void serviceSearchCompleted(int transID, int respCode) {  
    System.out.println("Terminada la busqueda de servicios");  
    //Si encontramos un servicio, lo usamos para mandar el mensaje(todos los  
    //servicios que hemos buscado son de puerto serie)  
    if (servicios_encontrados.size() > 0) {  
        enviarMensaje(txtMsjEnviado.getString());  
        //}  
    } else {  
        //Si no encontramos ningun servicio de puerto serie  
        cliente.mostrarDispositivos();  
        pantalla.setCurrent(cliente);  
    }  
}  
}  
}
```

Figura 6.10 Midlet para interacción entre el Router y el Hardware de Control

6.3 Implementación para el Celular Cliente cercano.

6.3.1 Codificación del midlet para el Celular Cliente cercano.

6.3.1.1 Telemandos de interacción entre el Cliente cercano y el Router.

La interacción con el cliente cercano se lo hace a través de un midlet en el Cliente cercano (complementándose con el midlet del Router), mismo que lo podemos apreciar en las gráficas de las Figuras 6-11,6-12 y 6-13.

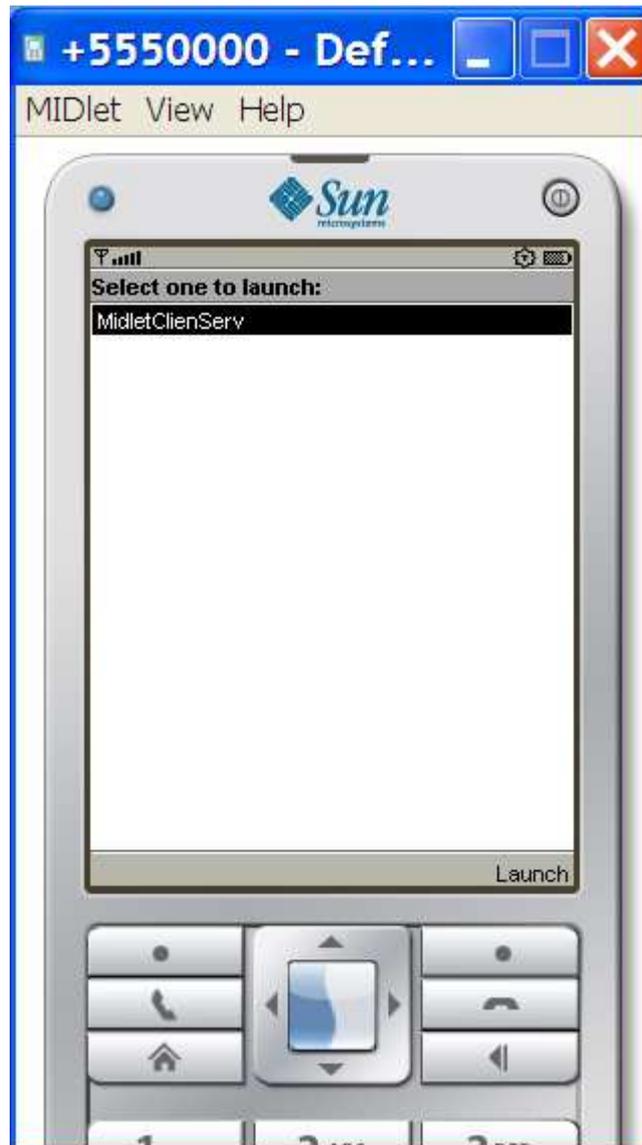


Figura 6-11 Ejecución del midlet en el Cliente cercano.

En la Figura 6-11, vemos la forma en la que inicia la ejecución de este Midlet MidletClienServ, denominado así porque contiene código que le permite comportarse



como cliente o servidor al mismo tiempo, y de esta manera poder enviar comandos de control al Router, o recibir alertas desde el Router.

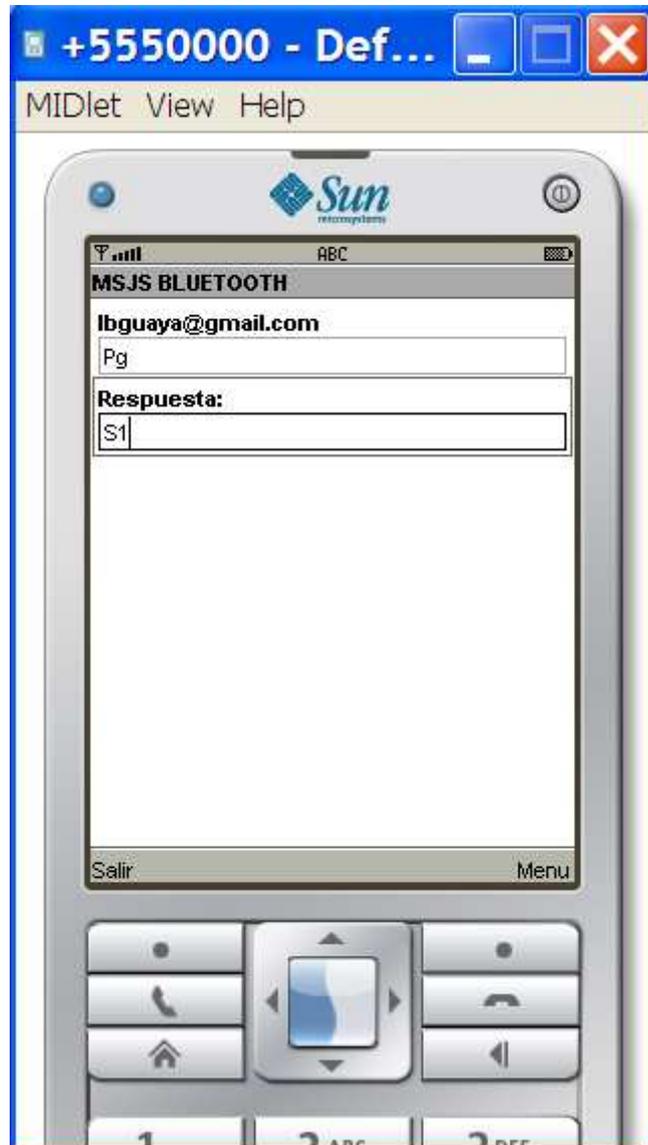


Figura 6-12 Envío y recepción de comandos al Router.

En la Figura 6-12, se muestra comandos en los cuadros de texto del simulador, que nos indican lo siguiente; que se ha enviado el comando Pg, para activar la puerta del garage desde el Hardware de Control, y en Respuesta vemos S1, que significa que el Sensor 1 del Hardware de Control ha sido activado.

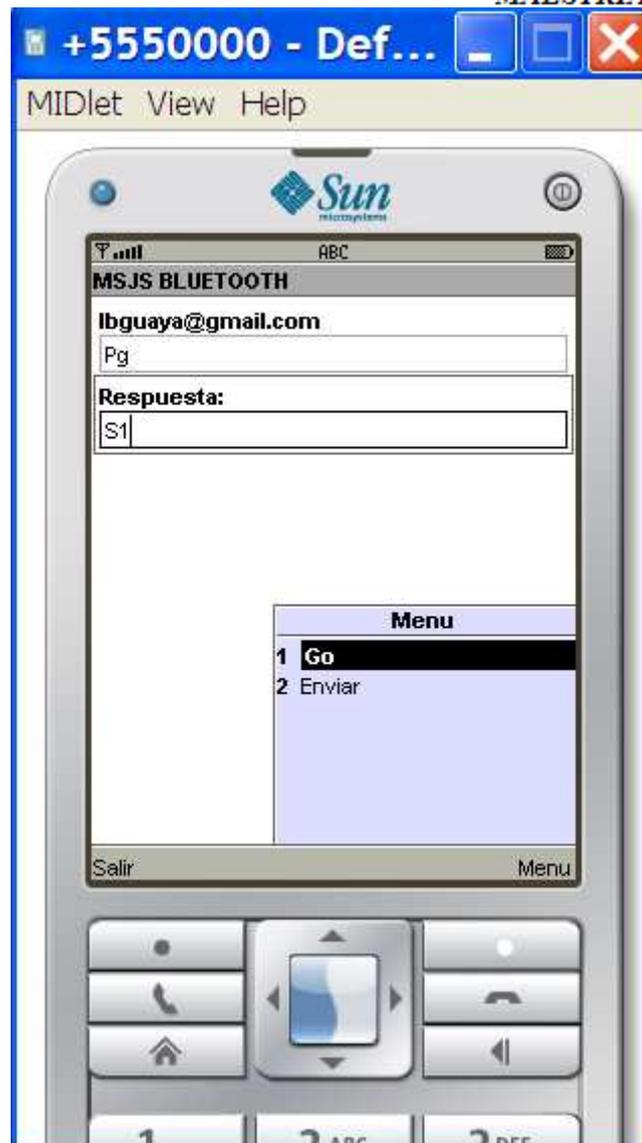


Figura 6-13 GUI para el Cliente cercano mostrando el menú de búsqueda y envío.

En la gráfica de la Figura 6-13, vemos que ya se ha enviado el comando Pg, vemos además un menú; con Go podemos iniciar la búsqueda de dispositivos, búsqueda de servicios, el establecimiento de la conexión y finalmente en la fase de comunicación es en donde ya se podría enviar nuevamente el comando Pg, al hacerlo, estamos teniendo control sobre el estado del pin en el Hardware de Control para cerrar la puerta del garage, recordemos que cada vez que manipulamos los pines para actuadores en el Hardware de Control, la primera vez activamos, luego desactivamos y así sucesivamente (activación y desactivación del pin del PIC).



CAPÍTULO 7: PRESUPUESTO.

7.1 Presupuesto.

Gastos de hardware

| Componente | Precio | Cantidad | Total |
|-------------------------------|--------|----------|-------|
| Diodos LED | 0,05 | 16 | 0,8 |
| Resistencia fija | 0,03 | 21 | 0,63 |
| PIC16877A | 6,6 | 1 | 6,6 |
| Batería de 9v de carbón | 1 | 1 | 1 |
| Pulsadores | 0,05 | 5 | 0,25 |
| Cristal de 4 MHz | 0,5 | 1 | 0,5 |
| Interruptores | 0,5 | 2 | 1 |
| Bornera de 2 pines | 0,2 | 1 | 0,2 |
| Regulador LM7805 | 0,5 | 1 | 0,5 |
| Regulador LM7833 | 0,5 | 1 | 0,5 |
| Capacitor ceramico 33pF | 0,1 | 2 | 0,2 |
| Capacitor electrolítico 100uF | 0,15 | 1 | 0,15 |
| RN-41 | 65 | 1 | 65 |
| Fibra de vidrio | 1 | 1 | 1 |
| Broche para batería | 0,15 | 1 | 0,15 |
| Regleta maquinada hembra | 0,5 | 1 | 0,5 |



| | | | |
|-------------------------|------|---|----------------|
| Percloruro férrico 100g | 0,5 | 1 | 0,5 |
| Suelda | 0,5 | 1 | 0,5 |
| Pasta de soldar | 0,25 | 1 | 0,25 |
| | | | \$80,23 |

Tabla 7-1 Gastos de hardware.

En la Tabla 7-1 observamos a detalle los gastos de implementación de hardware, que son 80,23 dólares americanos, esto se gasta en la compra de todos los componentes para la implementación del mainboard del hardware del sistema.

| Gastos de implementación | | | |
|---------------------------------|---------------|-----------------|-----------------|
| Concepto | Precio | Cantidad | Total |
| Dominio | 11,88 | 1 | 11,88 |
| Hosting | 56 | 1 | 56 |
| Chip PORTA | 7 | 2 | 14 |
| Targetas PORTA | 3 | 10 | 30 |
| Mes de Internet | 32 | 6 | 192 |
| Celular | 250 | 2 | 500 |
| | | | \$803,88 |

Tabla 7-2 Gastos de implementación.

La Tabla 7-2 muestra más gastos de implementación, aquí constan los gastos por compra de dos celulares, uno para que haga el papel de Cliente Lejano o Cliente Cercano y otro para el Router del sistema, ha sido llamado Router porque une la red GPRS de Porta con dos o más redes Bluetooth.



| Mano de obra por seis meses de trabajo | | | | |
|--|---------------|--------------|------------------------|---------------|
| Valor de hora ingeniería | Nro de Horas | Valor diario | Días de trabajo/semana | Valor semanal |
| \$10,00 | 5 | \$50,00 | 5 | \$250,00 |
| | | | | |
| Nro Semanas del mes | Valor mensual | Nro meses | Total | |
| 4 | \$1.000,00 | 6 | \$6.000,00 | |

Tabla 7-3 Gastos de mano de obra.

La Tabla 7-3 muestra el cálculo de la mano de obra de un desarrollador por seis meses de trabajo, muestra el valor de hora de ingeniería de 10 dólares que considero es el mínimo pago por hora de trabajo para alguien que desarrolle, integre, e implemente hardware y software en un sistema de éste tipo.

| Gasto Total | |
|--------------------------|-------------------|
| Gastos en hardware | \$80,23 |
| Gastos de implementación | \$803,88 |
| Mano de obra | \$6.000,00 |
| | |
| Subtotal | \$6.884,11 |
| Imprevistos 5% | \$344,21 |
| Total Proyecto | \$7.228,32 |

Tabla 7-4 Gasto total.



En la Tabla 7-4, muestra que como gasto total del sistema tenemos \$7.228,32 , se debe indicar que éste es el costo de desarrollo, pero una vez desarrollado el primer prototipo, el resto ya no requiere de todo este rubro.

Ahora de los \$7.228,32 determinemos cuáles son gastos fijos y cuáles recurrentes.

Como gasto recurrente tenemos \$80.23 de los gastos en hardware más \$500 de dos celulares (uno para el Router, y otro que puede hacer de Cliente cercano o lejano, ya que como mínimo deben haber 2 móviles para poder poner en funcionamiento el sistema), esto quiere decir que 580.23 es el costo del hardware del proyecto, y es el costo que se requerirá por cliente.

Como gasto fijo tenemos \$6648,09, que es la diferencia del gasto total del proyecto y los gastos recurrentes.



CAPÍTULO 8: CONCLUSIONES Y RECOMENDACIONES.

8.1 Conclusiones.

- Se ha logrado la transmisión de comandos e imágenes del celular Router al servidor Apache Tomcat con la ayuda de complejos principios de programación como lo son: threads, codificación de la imagen enviada, servlets, conexiones entre midlets y aplicaciones web.
- Se ha conseguido la implementación del sistema mediante la perfecta combinación entre sistemas de telecomunicaciones y sistemas informáticos para conseguir un versátil y moderno sistema de telemedición y telecontrol.
- Para la transmisión de imágenes se ha usado el algoritmo Base-64, el mismo que cifra la información, pero incrementado el tamaño de la imagen.
- La autenticación del sistema se la ha conseguido mediante una base de datos implementada en el servidor, permitiendo de esta manera que sólo usuarios autorizados manipulen el sistema.
- La implementación de la configuración local como remota, permite tener un eficiente y eficaz control de usuarios.
- El estado de los dispositivos es almacenado en una base de datos en el Servidor, consiguiendo de esta manera que todo acceso al hardware sea bajo la autenticación de usuarios desde el Servidor.



8.2 Recomendaciones.

- Recomiendo el uso de diagramas de flujo, como una práctica y sencilla herramienta que nos permite comprender y documentar la lógica de resolución del problema.
- Se recomienda fabricar un PCB independiente para el RN-41, y luego insertarlo en el mainboard del Hardware de Control mediante un zócalo para darle firmeza y facilidad de manipulación de este delicado componente.
- Al momento de contratar el hosting, verificar que el servidor sea Apache Tomcat, ya que las funcionalidades del Tomcat permite la ejecución de aplicaciones web de Java (.war).
- Verificar que en el datasheet del móvil que se adquiriera tenga soporte (jsrs) para Bluetooth, páginas HTML, y para captura de imágenes.
- El uso de servlets para la comunicación cliente-servidor, ya que nos permite implementar comunicaciones bidireccionales.
- El uso de threads para optimizar el funcionamiento del sistema a través de procesos concurrentes.



BIBLIOGRAFÍA.

Automatismos para puertas de garage.

http://www.construnario.com/notiweb/titulares_resultado.asp?regi=13280.

BricoGeek.com. *Sensor de movimiento PIR*. <http://www.bricogeek.com/shop/42-sensor-de-movimiento-pir.html>.

Cerradura eléctrica FE-789 Llave Plain Key (Multipunto).

<http://www.soprano.com.br/esp/produtos-internas.php?idProduto=391&idSegmento=59&idUnidade=1>.

DEITEL, Paul J. y Harvey M. DEITEL. *JAVA Cómo Programar*. Séptima edición. PEARSON EDUCACIÓN. ISBN:978-970-26-1190-5, 2008.

Electronics, Labcenter. *Software de Diseño de PCBs Proteus*. <http://www.labcenter.com/index.cfm>.

ESPINOSA, Julian. LÓPEZ, Óscar. GARCÍA, Santiago. *Técnico en Telecomunicaciones, Tomo III*. Madrid-España: Cultural, S.A. ISBN: 84-8055-560-2, 2002.

—. *Técnico en Telecomunicaciones, Tomo III*. Madrid-España: Cultural, S.A. ISBN: 84-8055-560-2, 2002.

GIBSON, Jerry. *The Mobile Communications Hand Book*. New York: CRC Press LLC, Second Edition, 1999.

HERNANDO, José María. *Comunicaciones Móviles*. Madrid-España: Centro de Estudios Ramón Areces, S.A. ISBN:84-8004-635-X, 2004.

Herrera, Luis. «Smart (Domotic) houses.» *Revista Ingeniería e Investigación* (58)., 2005: VOL. 25 No 2. 47-53.

Iluminación.

<http://www.casadomo.com/noticiasDetalle.aspx?c=145&m=21&idm=157&pat=20&n2=20>.



KNUDSEN, Jonathank and Sing Li. *Beginning J2ME From Novice to Professional*.

USA: ISBN: 1-59059-479-7, 2005.

Lámpara incandescente. <http://es.wikipedia.org/wiki/Ampolleta>.

Luminaria fluorescente. http://es.wikipedia.org/wiki/L%C3%A1mpara_fluorescente.

Mecanique. *MicroCode Studio*. <http://www.mecanique.co.uk/code-studio/>.

Microchip. *PIC16F87XA Data Sheet. 28/40/44-Pin Enhanced Flash*. U.S.A., 2003.

Puertas y ventanas.

<http://www.casadomo.com/noticiasDetalle.aspx?c=150&m=21&idm=161&pat=20&n2=20>.

ROMERO, M.A. *Domótica*. 1998.

RovingNetworks. *Wireless for less*. www.rovingnetworks.com.

SOLÁNS, Domingo. *Las Nuevas Tecnologías al Servicio de los Mayores DOMOTICA*. Castellón de la Plana, 2005.

THOMPSON, Tymoty. *Bluetooth Application Programing with the Java Apis. Essential Edition*. USA: ISBN: 978-0-12-374342-8, 2008.



INDICE DE FIGURAS

Figura 2-1. Sistema telemático aplicado a domótica.

Figura 2-2. Plataformas Java. (THOMPSON 2008)

Figura 2-3. Componentes de la Arquitectura Java ME. (THOMPSON 2008)

Figura 2-4. PIC16F877A. (Microchip 2003)

Figura 2-5 Sensor de Movimiento. (BricoGeek.com s.f.)

Figura 2-6 Cerradura Eléctrica FE 789. (Cerradura eléctrica FE-789 Llave Plain Key (Multipunto) s.f.)

Figura 3-1 Análisis del problema.

Figura 3-2 Análisis del hardware que interactúa con el router.

Figura 3-3 Diagrama de flujo para el PIC.

Figura 3-4 Simulación del hardware con Proteus.

Figura 3-5 Diseño del PCB para el RN-41.

Figura 3-6 Diseño del PCB del hardware.

Figura 3-7 Cargando comandosbluetooth.hex en el PIC (1).

Figura 3-8 Cargando comandosbluetooth.hex en el PIC (2).

Figura 3-9 Hardware del sistema (frontal).

Figura 3-10 Hardware del sistema (reverso).

Figura 3-11 Probando el hardware con la hiperterminal.

Figura 3-12 Probando salidas para los actuadores.

Figura 3-13 Probando entradas para los sensores.

Figura 4-1 Análisis del problema: Interacción cercana.

Figura 4-2 Análisis del problema: Interacción lejana.

Figura 4-3 Análisis del problema: Registro de usuarios lejano y cercano.

Figura 5-1 Caso de uso Telecontrol cercano.

Figura 5-2 Caso de uso Telemedición cercana.

Figura 5-3 Caso de uso Telecontrol lejano.



Figura 5-4 Caso de uso Telemedición lejana.

Figura 5-5 Caso de uso Registro lejano.

Figura 5-6 Caso de uso Registro cercano.

Figura 6-1 Imagen capturada desde el Router.

Figura 6-2 Base de Datos del Sistema.

Figura 6-3 Tabla cliente de la Base de Datos.

Figura 6-4 Tabla datos de la Base de Datos.

Figura 6-5 Tabla dispositivos de la Base de Datos.

Figura 6-6 Propiedades del Router.

Figura 6-7 Simulando la captura de imágenes desde el Router.

Figura 6-8 Código fuente en J2ME de la clase que envía alertas al cliente lejano.

Figura 6-9 Clase para interacción entre el Router y el Hardware de Control.

Figura 6-10 Midlet para interacción entre el Router y el Hardware de Control.

Figura 6-11 Ejecución del midlet en el Cliente cercano.

Figura 6-12 Envío y recepción de comandos al Router.

Figura 6-13 GUI para el cliente cercano mostrando el menú de búsqueda y envío.



INDICE DE TABLAS

Tabla 2-1. J2ME Configuraciones, Perfiles, y APIs Opcionales. (KNUDSEN 2005)

Tabla 2-2. Comparación de Tecnologías de Comunicación Inalámbrica. (THOMPSON 2008).

Tabla 7-1 Gastos de hardware.

Tabla 7-2 Gastos de implementación.

Tabla 7-3 Gastos de mano de obra.

Tabla 7-4 Gasto total.



GLOSARIO DE TERMINOS Y ABREVIATURAS

Router: Se usó este término para indicar que sobre un móvil celular se implementó la funcionalidad de un router, que es unir dos o más redes.

Cliente cercano: Usuario capaz de realizar un control local (Bluetooth).

Cliente lejano: Usuario capaz de realizar un control remoto (GPRS).

Servidor: Computador que aloja un servlet (para nuestro caso particular).

Hardware de Control: Hardware al cuál se conectan los sensores y actuadores.

J2ME: Java 2 Micro Edition, que es un tecnología orientada al desarrollo de aplicaciones para dispositivos móviles.

Servlet: Aplicación que se ejecuta en el servidor, y tiene la capacidad de recibir peticiones y enviar respuestas, desde y hasta el cliente respectivamente.

VCD: Voltios de Corriente Directa.

Bps: Bits por segundo, que es el número de impulsos elementales (1 ó 0) transmitidos en cada segundo.

Dominio: Nombre que permite asociar a una IP, y acceder a dicha dirección. Es decir es el nombre que identifica a un sitio web.

Hosting: Servicio en el cual un proveedor alquila su computador para alojar un contenido accesible vía web. Es decir es el servicio que provee a los usuarios de Internet un sistema para poder almacenar información, imágenes, video, o cualquier contenido accesible vía web.

SGBD: Sistema Gestor de Base de Datos, es un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

PIC: Significa Peripheral Interface Controller, es decir es un controlador de periféricos.

CLDC: Connected, Limited Device Configuration.

MIDP: Mobile Information Device Profile.

Symbian: Sistema operativo muy difundido en dispositivos móviles.

Domótica: Inteligencia para domicilios.

SMS: Send Message Short.



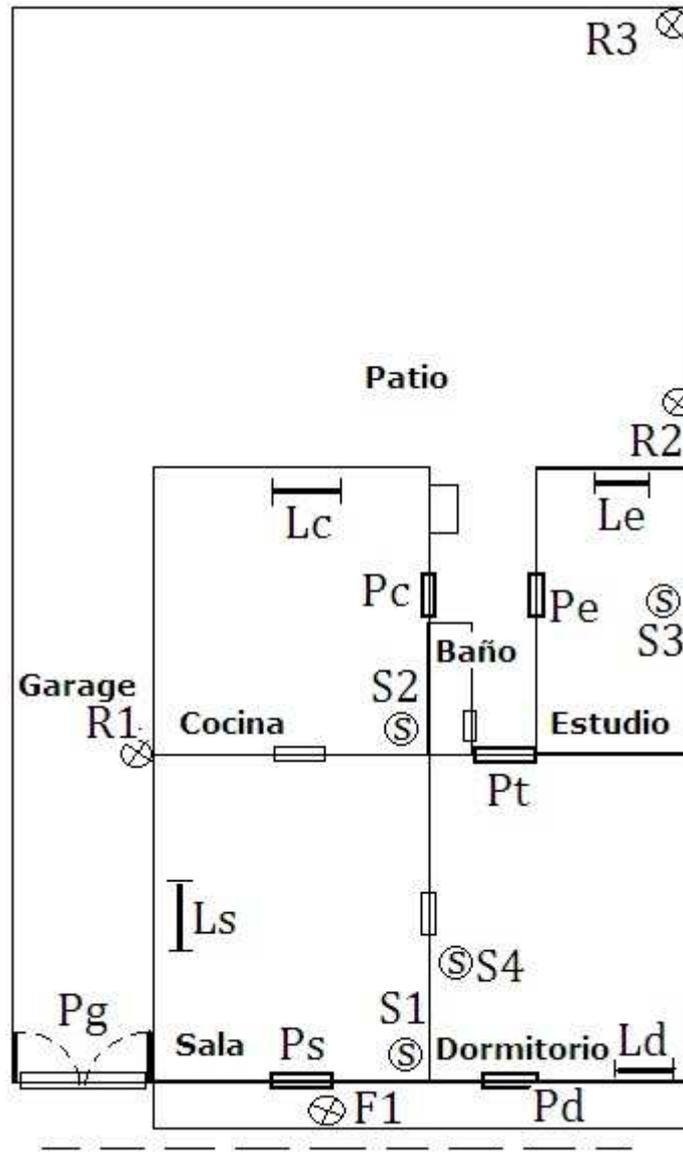
Sistema telemático: Sistema que une técnicas de Electrónica y Comunicaciones, Computación e Informática.

IDE: Entorno de Desarrollo Integrado. Programa informático compuesto por un conjunto de herramientas de programación.

SDK: Kit de Desarrollo de Software. Conjunto de herramientas de desarrollo que le permite a un programador crear aplicaciones para un sistema concreto.

ANEXOS.

A-1. Plano de la vivienda a automatizar.



En la figura 1, observamos el plano de un domicilio con 18 dispositivos, de los cuales 14 son actuadores y 4 son sensores, mismos que requieren ser controlados desde un teléfono móvil celular vía GPRS y Bluetooth. Donde Px, son cerraduras eléctricas de 12v, Rx son reflectores de 150w, Lx son lámparas fluorescentes de 20 w, F1 es una lámpara fluorescente ahorradora de 23 w y Sx son sensores de movimiento (PIR).

A-2. Hoja de Especificaciones del Módulo Bluetooth RN-41. (RovingNetworks s.f.)



RN-41

www.rovingnetworks.com

DS-RN41-V2.3 1/16/2009

Class 1 Bluetooth® 2.1 Module



Features

- Fully qualified Bluetooth 2.1/2.0/1.2/1.1 module
- Bluetooth v2.0+EDR support
- Postage stamp sized module, 13.4mm x 25.8 mm x 2mm
- UART (SPP or HCI) and USB (HCI only) data connection interfaces.
- Sustained SPP data rates - 240Kbps (slave), 300Kbps (master)
- HCI data rates - 1.5Mbps sustained, 3.0Mbps burst in HCI mode
- 8MB on board flash, HCI mode, or SPP/DUN software stacks available.
- Embedded Bluetooth stack profiles included (requires no host stack): GAP, SDP, RFCOMM and L2CAP protocols, with SPP and DUN profile support.
- Bluetooth SIG Qualified, End Product Listing
- Castellated SMT pads for easy and reliable PCB mounting

Applications

- Cable replacement
- Barcode scanners
- Measurement and monitoring systems
- Industrial sensors and controls
- Medical devices
- Asset tracking

Description

The RN41 provides a small form factor, low power, highly economic Bluetooth solution for OEM's adding wireless capability to their products. The RN41 supports multiple interface protocols, is simple to design in and fully certified, making it a complete embedded Bluetooth solution. By supporting the Enhanced Data Rate (EDR) Bluetooth® specification, the RN41 delivers three times the data rate than v1.2 Bluetooth devices. Designers can customize their application using the external interface to access up to 8Mbits of flash memory. The RN41 is the perfect product for developers who want to add wireless

Overview

- Baud rate speeds: 1200bps up to 921Kbps, non-standard baud rates can be programmed.
- Class 1 radio, 330' (100m) distance, 12dBm output transmitter, -80dBm typical receive sensitivity
- Frequency 2402 ~ 2480MHz.
- FHSS/GFSK modulation, 79 channels at 1MHz intervals
- Secure communications, 128 bit encryption
- Error correction for guaranteed packet delivery
- UART local and over-the-air RF configuration
- Auto-discovery/pairing requires no software configuration (instant cable replacement).
- Auto-connect master, IO pin (DTR) and character based trigger modes

Environmental Conditions

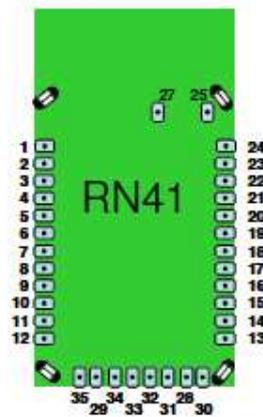
| Parameter | Value |
|-------------------------------|----------------|
| Temperature Range (Operating) | -40 °C ~ 85 °C |
| Temperature Range (Storage) | -40 °C ~ 85 °C |
| Relative Humidity (Operating) | ≤90% |
| Relative Humidity (Storage) | ≤90% |

A-3. Pinout del Módulo Bluetooth RN-41. (RovingNetworks s.f.)



DS-RN41-V2.3 1/16/2009

Pin Description



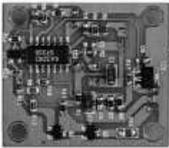
Top view

| Pin | Name | Description | Default |
|-------|----------|--|----------------------------------|
| 1 | GND | | |
| 2 | SPI MOSI | Programming only | No Connect |
| 3 | PIO6 | IO (HIGH-AUTO MASTER) | Input to RN41 with weak pulldown |
| 4 | PIO7 | IO (HIGH- force 9600 baud), low - 115K baud | Input to RN41 with weak pulldown |
| 5 | RESET | Active LOW reset optional | Input to RN41 with 1K pullup |
| 6 | SPI CLK | Programming only | No Connect |
| 7 | PCM CLK | PCM interface | No Connect |
| 8 | PCM SYNC | PCM interface | No Connect |
| 9 | PCM IN | PCM interface | No Connect |
| 10 | PCM OUT | PCM interface | No Connect |
| 11 | VDD | 3.3V regulated Power In | |
| 12 | GND | | |
| 13 | UART_RX | UART Receive Input | Input to RN41 with pullup |
| 14 | UART_TX | UART Transmit output | High level output from RN41 |
| 15 | UART_RTS | UART RTS out , goes HIGH to disable host transmitter | Low level output from RN41 |
| 16 | UART_CTS | UART CTS input, if set HIGH, disables transmitter | Low level input to RN41 |
| 17 | USB_D+ | USB port | Pull up 1.5K when active |
| 18 | USB_D- | USB port | |
| 19 | PIO2 | High when connected, Low otherwise | Output from RN41 |
| 20 | PIO3 | Auto discovery - HIGH | Input to RN41 with weak pulldown |
| 21 | PIO5 | Status: toggles based on state, LOW on connect | Output from RN41 |
| 22 | PIO4 | To set Factory defaults start HIGH, then toggle 3x | Input to RN41 with weak pulldown |
| 23 | SPI_CSB | Programming only | No Connect |
| 24 | SPI_MISO | Programming only | No Connect |
| 25 | GND | | |
| 26 | NC | RF pad keep all traces and planes clear. | |
| 27-29 | GND | | |
| 30 | AIO1 | Analog AD input, 0-1.8VDC 8 bit. | Charger monitor |
| 31 | PIO8 | IO (RF data/config LED option) | Output from RN41 |
| 32 | PIO9 | IO | Input to RN41 with weak pulldown |
| 33 | PIO10 | IO (remote DTR signal) | Input to RN41 with weak pulldown |
| 34 | PIO11 | IO (remote RTS signal) | Input to RN41 with weak pulldown |
| 35 | AIO1 | Analog AD input, 0-1.8VDC 8 bit. | Battery monitor |

809 University Avenue • Los Gatos, CA 95032 • Tel (408) 395-6539 • info@RovingNetworks.com

A-4. Hoja de Especificaciones del Sensor de Movimiento. (BricoGeek.com s.f.)

SE-10



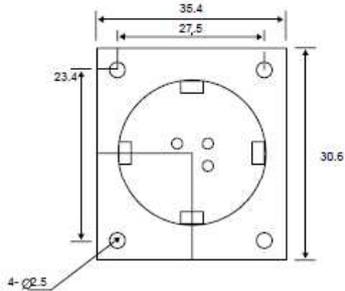
• BACK SIDE





• FRONT SIDE

Dimensions



Specifications

| ITEM | | Specification | Unit | Condition |
|-----------------------|--------------|-----------------|-------------|---|
| Sensor Type | | Dual Element | | |
| Housing | | TO 5 | | |
| Element Size | | 2×1 | mm | |
| Spacing | | 1 | mm | |
| Responsivity | Min | 3.2 | xv/w | 7...14mm, 1Hz, 100°C (One element cover) |
| | Typ | 4.0 | | |
| Match | Max | <10 | % | 7...14mm, 1Hz, 100°C (Both element expose) |
| | | | | |
| Noise | Typ | 20 | iV/p-p V | 25°C, 0.4...10Hz |
| | Max | 50 | | |
| Effect Voltage | Min | 0.2 | | Re=47XO |
| | Max | 1.5 | | |
| Window Material | | Silicon, coated | | |
| Spectral Range | Transmission | T>30 average | % | 7...14mm <5mm |
| | Blocking | T<0.1 | | |
| Operating Voltage | | 12 | V | |
| Operating Temperature | | -10~40 | °C | |
| Storage Temperature | | -40~80 | °C | |



A-5. Código fuente en PBPv2.44, editado con MicrocodeStudio v3.005.

```
'* Nombre: COMANDOSBLUETOOTH.BAS *
'* Autor : ING. LENIN BLADIMIR GUAYA DELGADO *
'* Aviso : Copyright (c) 2010 *
'* : All Rights Reserved *
'* Fecha : 05/07/2010 *
'* Versión : 1.0 *
'* Notes : Firmware para envío y recepción de comandos vía *
'* : Bluetooth *
```

```
define OSC 4 ;Indico que usaré un cristal de 4MHz

INCLUDE "modedefs.bas" ;Incluyendo los modos de comunicación serial asíncrona

MENSAJE2 VAR Byte[4] ;Variable que almacena los comandos entrantes

TRISB =0 ;Seteo del PORTB para salida

TRISC = 0 ;Seteo del PORTC para salida

TRISD=0 ;Seteo del PORTD para salida

' Seteo para trabajar por pin, no por puerto

Portb.0=1'Para cambio de mode, para futuras funcionalidades.

;*18 dispositivos: 14 actuadores y 4 sensores.
```



Portb.1=0 ;*1

Portb.2=0 ;*2

Portb.3=0 ;*3

Portb.4=0 ;*4

Portb.5=0 ;*5

Portb.6=0 ;*6

Portb.7=0 ;*7

Portc.0=0 ;*8

Portc.1=0 ;*9

Portc.2=0 ;*10

Portc.3=0 ;*11

Portd.0=0 ;*12

Portd.1=0 ;*13

Portd.2=0 ;*14

Portd.3=0 'Se setea a 0 porque es una salida para LED indicador de inicio.

Portd.4=1 ;*1

Portd.5=1 ;*2

Portd.6=1 ;*3

Portd.7=1 ;*4

PORTD.3 = 1; Encendido led indicador de inicio

pause 1500



PORTD.3 = 0; Apagado led indicador de funcionamiento

programablutooth:

SERIN2 PORTC.7,84,[skip(0),str mensaje2\2] ;Leo los datos que le llegan al

;RN-41

If (mensaje2[0]="P") and (mensaje2[1]="g") AND (PORTB.1=0)THEN Pg;Ir a func

If (mensaje2[0]="P") AND (mensaje2[1]="g") AND (PORTB.1=1)THEN Pg2:

if (mensaje2[0]="P") AND (mensaje2[1]="s") AND (PORTB.2=0)THEN Ps:

if (mensaje2[0]="P") AND (mensaje2[1]="s") AND (PORTB.2=1)THEN Ps2:

if (mensaje2[0]="P") AND (mensaje2[1]="c") AND (PORTB.3=0)THEN Pc:

if (mensaje2[0]="P") AND (mensaje2[1]="c") AND (PORTB.3=1)THEN Pc2:

if (mensaje2[0]="P") AND (mensaje2[1]="e") AND (PORTB.4=0)THEN Pe:

if (mensaje2[0]="P") AND (mensaje2[1]="e") AND (PORTB.4=1)THEN Pe2:

if (mensaje2[0]="P") AND (mensaje2[1]="t") AND (PORTB.5=0)THEN Pt:

if (mensaje2[0]="P") AND (mensaje2[1]="t") AND (PORTB.5=1)THEN Pt2:

if (mensaje2[0]="P") AND (mensaje2[1]="d") AND (PORTB.6=0)THEN Pd:

if (mensaje2[0]="P") AND (mensaje2[1]="d") AND (PORTB.6=1)THEN Pd2:

if (mensaje2[0]="L") AND (mensaje2[1]="s") AND (PORTB.7=0)THEN Ls:

if (mensaje2[0]="L") AND (mensaje2[1]="s") AND (PORTB.7=1)THEN Ls2:

if (mensaje2[0]="L") AND (mensaje2[1]="c") AND (PORTC.0=0)THEN Lc:

if (mensaje2[0]="L") AND (mensaje2[1]="c") AND (PORTC.0=1)THEN Lc2:

If (mensaje2[0]="L") AND (mensaje2[1]="e") AND (PORTC.1=0)THEN Le:

If (mensaje2[0]="L") AND (mensaje2[1]="e") AND (PORTC.1=1)THEN Le2:



if (mensaje2[0]="L") AND (mensaje2[1]="d") AND (PORTC.2=0)THEN Ld:

if (mensaje2[0]="L") AND (mensaje2[1]="d") AND (PORTC.2=1)THEN Ld2:

if (mensaje2[0]="R") AND (mensaje2[1]="1") AND (PORTC.3=0)THEN
Reflector1:

if (mensaje2[0]="R") AND (mensaje2[1]="1") AND (PORTC.3=1)THEN
Reflector12:

if (mensaje2[0]="R") AND (mensaje2[1]="2") AND (PORTD.0=0)THEN
Reflector2:

if (mensaje2[0]="R") AND (mensaje2[1]="2") AND (PORTD.0=1)THEN
Reflector22:

if (mensaje2[0]="R") AND (mensaje2[1]="3") AND (PORTD.1=0)THEN
Reflector3:

if (mensaje2[0]="R") AND (mensaje2[1]="3") AND (PORTD.1=1)THEN
Reflector32:

if (mensaje2[0]="F") and (mensaje2[1]="1") AND (PORTD.2=0)THEN Foco1:

if (mensaje2[0]="F") and (mensaje2[1]="1") AND (PORTD.2=1)THEN Foco12:

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablutooth:

sensores:

If (PORTD.4=0)THEN S1: ;Si se activa el S1, voy hasta la función que envía

If (PORTD.5=0)THEN S2: ;datos al puerto.

If (PORTD.6=0)THEN S3:

If (PORTD.7=0)THEN S4:



goto programablueetooth:

'Entradas de los 4 sensores de movimiento

S1:

serout2 PORTC.6,84,["S1",13] ; Envío de datos al celular router

PORTD.4= 1 ;Restableciendo el estado del pin de entrada.

goto programablueetooth:

S2:

serout2 PORTC.6,84,["S2",13] ; Envío de datos al celular router

PORTD.5= 1 ;Restableciendo el estado del pin de entrada.

goto programablueetooth:

S3:

serout2 PORTC.6,84,["S3",13] ; Envío de datos al celular router

PORTD.6= 1 ;Restableciendo el estado del pin de entrada.

goto programablueetooth:

S4:

serout2 PORTC.6,84,["S4",13] ; Envío de datos al celular router

PORTD.7= 1 ;Restableciendo el estado del pin de entrada.

goto programablueetooth:

'Activación de pines para los 14 actuadores

Pg:



PORTB.1=1 ;Activo pin 34 para abrir la puerta del garage

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:
;Cambio

; de estado al pin, y luego verifico alguna alarma desde los sensores.

goto programablueetooth:

Pg2:

PORTB.1=0 ;Desactivo pin 34 para cerrar la puerta del garage

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Ps:

PORTB.2=1 ;Activo pin 35 para abrir la puerta de la sala

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Ps2:

PORTB.2=0 ;Desactivo pin 35 para cerrar la puerta de la sala

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Pc:

PORTB.3=1 ;Activo pin 36 para abrir la puerta de la cocina

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Pc2:

PORTB.3=0 ;Desactivo pin 36 para cerrar la puerta de la cocina



IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Pe:

PORTB.4=1 ;Activo pin 37 para abrir la puerta del estudio

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Pe2:

PORTB.4=0 ;Desactivo pin 37 para cerrar la puerta del estudio

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Pt:

PORTB.5=1 ;Activo pin 38 para abrir la puerta trasera

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Pt2:

PORTB.5=0 ;Desactivo pin 38 para cerrar la puerta trasera

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Pd:

PORTB.6=1 ;Activo pin 33 para abrir la puerta del dormitorio

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:



Pd2:

PORTB.6=0 ;Desactivo pin 33 para cerrar la puerta del dormitorio

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Ls:

PORTB.7=1 ;Activo pin 39 para encender la lámpara de la sala

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Ls2:

PORTB.7=0 ;Desactivo pin 39 para apagar la lámpara de la sala

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Lc:

PORTC.0=1 ;Activo pin 39 para encender la lámpara de la cocina

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Lc2:

PORTC.0=0 ;Desactivo pin 39 para apagar la lámpara de la cocina

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Le:

PORTC.1=1 ;Activo pin 15 para encender la lámpara del estudio



IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Le2:

PORTC.1=0 ;Desactivo pin 15 para apagar la lámpara del estudio

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Ld:

PORTC.2=1 ;Activo pin 16 para encender la lámpara del dormitorio

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Ld2:

PORTC.2=0 ;Desactivo pin 16 para apagar la lámpara del dormitorio

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Reflector1:

PORTC.3=1 ;Activo pin 17 para encender el reflector 1

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Reflector12:

PORTC.3=0 ;Desactivo pin 17 para apagar el reflector 1

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:



Reflector2:

PORTD.0=1 ;Activo pin 18 para encender el reflector 2

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Reflector22:

PORTD.0=0 ;Desactivo pin 18 para apagar el reflector 2

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Reflector3:

PORTD.1=1 ;Activo pin 19 para encender el reflector 3

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Reflector32:

PORTD.1=0 ;Desactivo pin 19 para apagar el reflector 3

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Foco1:

PORTD.2=1 ;Activo pin 20 para encender el foco 1

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

Foco12:

PORTD.2=0 ;Desactivo pin 20 para apagar el foco 1



UNIVERSIDAD DE CUENCA
FACULTAD DE INGENIERÍA
MAESTRÍA EN TELEMÁTICA

IF ((PORTD.4=0)or(PORTD.5=0)or(PORTD.6=0)or(PORTD.7=0))then sensores:

goto programablueetooth:

END



A-6. Verificación de la disponibilidad del dominio en la Web.

Fast Domain Name Search | Domize - Mozilla Firefox

Archivo Editar Ver Historial Delicioso Marcadores Herramientas Ayuda

https://domize.com/

HOME NEWS & BUZZ ABOUT FAQ PRO

DOM!ZE

Start typing a domain name below: (e.g. for the website www.abc.com you

ALL COM AVAILABLE (3) ANY AVAILABLE (4) * STARRED (0)

Results
Available (blue) domains click through to GoDaddy for purchase ([change](#))

| | REG'D | 2010 | 04 | 07 | EXP | 2011 | 04 | 07 |
|----------|-------|------|-----|-----|-----|------|----|----|
| lenguaya | com | net | org | biz | | | | |
| lenguay | com | net | org | biz | | | | |
| lengu | com | net | org | biz | | | | |

Copyright © 2010 [Rare Window](#) | [Follow us](#) on the Twitter | [Privacy Information](#)
Put Domize on your blog or website with our [widget](#) | SSL by GoDaddy, certificates for \$12.99/year

Terminado domize.com



A-7. Página web del proveedor del servicio de hosting.





A-8. Activación de la cuenta para el servicio de hosting.

=====

IMCOMEPRO Su Proveedor de Web Hosting – Activación de Cuenta

=====

Apreciado Usuario,

Gracias por confiar en **IMCOMEPRO** como su Proveedor de Web Hosting!

Nos complace informarle que su cuenta ya ha sido activada.

Fecha de Inicio: Viernes, 02 de Julio de 2.010

Por favor imprima esta Información para futuras referencias.

=====

Detalle de la Cuenta

=====

Lea las siguientes instrucciones para manejar su contenido y pueda interactuar con su Servicio de Web Hosting:

Su Cuenta de Alojamiento es: Hosting Pymes

El Usuario de su Cuenta es: **leningua**

El Password de su Cuenta es: **mat2010**

Su Cuenta de Dominio es: leninguaya.com



El Home Root de su Cuenta es: /home/**leningua**

El Servidor Perl es: /usr/bin/perl

El Servidor Sendmail es: /usr/sbin/sendmail

=====

Paso 1 – Información de los Name Servers:

=====

Por favor cambie los Name Servers de su Dominio a las siguientes direcciones, esto es solo para Usuarios que no hayan registrado su Dominio con **IMCOMEPRO**:

servidores.imcomepro.com (Primario)

dns77.imcomepro.com (Secundario)

Si usted gusta podemos realizar el cambio de los Name Servers por usted (sin costo), envíenos un Ticket de Soporte con la siguiente información:

- 1) Su nombre de Dominio
- 2) El Nombre del Registrador (donde usted registro su dominio)
- 3) Requerimos el Login para poder hacer el cambio de los DNS con su anterior registrador. (i.e. username y password)

Para que la propagación de su Dominio este completamente lista en Internet, requiere de 12 a 72 horas, dependiendo de su ISP.

Mientras espera que se realice la propagación de su Dominio en Internet, usted ya puede subir su contenido con su dirección IP, instalar sus scripts, configurar sus cuentas de Correo Electrónico para empezar a interactuar con su Servicio de Web Hosting.

=====



Paso 2 - Subir su contenido vía el FTP:

=====

Usted puede empezar a subir su contenido utilizando su username y password asignado en su detalle de la Cuenta.

Una vez propagado su Dominio en Internet, usted podrá subir sus archivos usando el nombre de Dominio, su username y password asignados en su detalle de la Cuenta.

Si no dispone de un programa **FTP** para subir sus archivos a Internet, le podemos proporcionar este por un costo de 8 USD incluido IVA.

Cuando ingrese al ftp de su cuenta, usted vera numerosas carpetas y directorios.

Le explicamos a continuación cuales son:

1) tmp

Esta es una sesión donde se crean los archivos en PHP y donde se guardan los scripts; por favor no suba ningún contenido en esta carpeta. Los archivos creados de las Estadísticas de sus visitantes están localizados en este directorio.

2) public_html

Esta es su carpeta root Web. En esta podrá subir el contenido de su Sitio Web, todos los archivos que desee mostrar en Internet deben estar en esta carpeta. Usted puede crear las subcarpetas que su sitio requiera.

3) public_ftp

Esta es su carpeta FTP anónima. Todos los archivos subidos por usuarios vía FTP anónima serán guardados en este directorio.

4) mail



Esta es la carpeta donde están localizadas sus cuentas de Correo Electrónico, carpetas y archivos (es decir el inbox, los mensajes enviados, etc).

5) etc

Este es el directorio donde están localizados algunos de sus archivos setting, por favor no suba contenido y no borre ningún archivo.

6) www

Este es un eslabón simbólico del directorio public_html, puede subir en está carpetas de contenido o usar scripts, es casi lo mismo, tú puedes ver las carpetas que estas usando en public_html.

=====

Paso 3 – Acceso a su Cuenta con el Panel de Control:

=====

Para acceder a su Panel de Control, por favor realice las siguientes instrucciones, incluya su username y password asignados en su detalle de la Cuenta.

En su Panel de Control tiene todas las opciones y características para generar un óptimo manejo y aprovechar al máximo su servicio de Web Hosting.

Después de la propagación en Internet de su Dominio, el acceso a su Panel de Control será: <http://cpanel.leninguaya.com>

Si tiene alguna inquietud sobre como usar su Panel de Control no dude en enviarnos un Ticket de Soporte

=====

Paso 4 - Conectado con su Servidor de Correo-Electrónico:

=====



Para conectar su Servidor de Correo Electrónico, por favor use la siguiente dirección para ambos: mail.leninguaya.com, esta dirección es el Server Hostname de Correo Electrónico entrante y saliente (POP3, IMAP, SMTP).

Recuerde que necesita activar la opción de (mi servidor requiere autenticación en Outlook Express).

Su Dominio leninguaya.com necesita estar propagado alrededor de Internet correctamente para acceder a su Servidor de Correo Electrónico, este detalle esta explicado en la sección de Name Servers.

Para el Acceso vía Web Mail diríjase a: correo.leninguaya.com

Para reducir la cantidad de transferencia mensual que utilizará, usted puede también usar su ISP Smtip Server – Su destinatario de Correo Electrónico lo verá de la misma forma.

=====

Paso 5 - Use el Certificado SSL Shared:

=====

Usar su Certificado SSL Shared (Secure Sockets Layer) es fácil, utilícelo solo en caso de requerirlo su sitio, usted solo necesita reemplazar su Dominio con:

<https://secure3.servidores.imcomepro.com/~username/>

Ejemplo: Si su dominio es [qwerty.com](http://www.qwerty.com), su username es qwerty1, si usted desearía asegurar la dirección <http://www.qwerty.com/orderform.html>, usted tendría que reemplazar www.qwerty.com con secure3.servidores.imcomepro.com/~qwerty1 y así

<https://secure3.servidores.imcomepro.com/~qwerty1/orderform.html> sería ahora y de esta manera una página segura.



Solo con **IMCOMEPRO** podrá tener una página segura para realizar transacciones en Línea por un costo muy reducido Consúltenos.

=====

Paso 6 - Para Soporte adicional:

=====

Una vez más, gracias por escogernos y le damos la bienvenida a **IMCOMEPRO** su Proveedor de Web Hosting!

En nuestro Sistema de Soporte, hay Operadores en Línea en determinadas horas del día, aquí debe enviar su Ticket de Soporte para cualquier departamento.

Recuerde que nuestro departamento de Soporte Técnico trabaja vía Internet las 24 horas del día.

Para ingresar al Sistema de Soporte Técnico digite:

http://www.imcomepro.com/php/c_soporte.php

Contáctenos para más información, a continuación le detallamos las direcciones de Correo Electrónico para los diferentes departamentos y poder atenderlo mejor:

Servicio al Cliente: info@imcomepro.com

Ventas Directas: ventas@imcomepro.com

Diseño Web: diseño@imcomepro.com

Registro de Dominios: registrador@imcomepro.com

Soporte Técnico: soporte@imcomepro.com



IMCOMEPRO su Proveedor de Web Hosting, Correo Electrónico y Registro de
Dominios

Quito – Ecuador

Recomiéndonos con sus amigos o colegas, visítenos en Internet digitando:

<http://www.imcomepro.com>

Cuenta Nueva de Web Hosting Activada:

=====

Nombre de Dominio....: leninguaya.com

Cuenta.....: Hosting Pymes

Ciclo de Pago.....: Anual

Código de Promoción...: Ninguno

Dirección de E-mail.....: lbguaya@gmail.com

Términos y Condiciones del Contrato...: Aceptadas

Hora & Fecha.....: 17:26:15 / 02-07-2010

A-9. Simulación del sistema telemático aplicado a la domótica.

