Diseño e implementación de una solución de monitoreo remoto vía Internet, para una red inalámbrica de sensores

Oswaldo Fernando Daqui Solano y Fabio M. González G.,
Escuela Politécnica Nacional (EPN), Quito – Ecuador

***Resumen*– El presente artículo muestra el desarrollo de un sistema de monitoreo vía Web para una red inalámbrica de sensores. Primero se programan los nodos sensores utilizando el sistema operativo TinyOS junto con el lenguaje de programación nesC, luego se envían las mediciones recolectadas a un nodo estación base conectado a un computador. Se interpreta y almacena las mediciones de los sensores a una base de datos MySQL para finalmente desarrollar el sistema de monitoreo en lenguaje PHP. El acceso al sistema es seguro y administrable, permite monitorear en tiempo real, hacer gráficas estadísticas y notificar alarmas vía Web, email o con mensajes de texto mediante un celular.**

***Índices* – IEEE 802.15.4, MySQL, monitoreo web, redes, inalámbricas, sensores, Internet, tiempo real, TinyOS, PHP.**

I. Introducción

Las redes cableadas de sensores no son nada nuevo y se las ha utilizado ampliamente para monitorear instalaciones importantes como plantas químicas, fábricas, etc. Sin embargo en los últimos años, y gracias al avance tecnológico han surgido nuevas tecnologías como son las redes inalámbricas de sensores, un área de investigación muy activa debido a su amplia gama de aplicaciones, no solamente en la ciencia y la ingeniería, sino también en la agricultura, atención de la salud, medio ambiente, seguridad y protección de infraestructuras críticas, calidad de vida, etc.

El desarrollo de Internet y las tecnologías circundantes han provocado un auge de múltiples aplicaciones web, así se pueden encontrar aplicaciones web en educación, multimedia, cartografía, etc. Dentro de este amplio espectro también se pueden hallar aplicaciones para monitoreo de redes de computadores personales, servidores, equipos de *networking*, sin embargo en temas de redes inalámbricas de sensores las soluciones de este tipo pueden llegar a ser muy costosas, debido a que es una tecnología relativamente nueva.

La solución propuesta en el presente trabajo busca contribuir con un sistema de monitoreo económico y seguro para las redes inalámbricas de sensores, integrando herramientas propias del sistema operativo TinyOS a una interfaz amigable que permita de una manera sencilla monitorear y analizar los valores medidos por los nodos sensores, ya sea en forma local o remota vía web.

II. Redes inalámbricas de sensores

Una red inalámbrica de sensores es un sistema autónomo compuesto de diminutos nodos equipados con sensores y capacidades de procesamiento [1]. Su reducido tamaño y capacidad de transmitir sin cables, permiten un despliegue rápido y flexible de centenares a miles de dispositivos. Estos nodos se han desarrollado gracias al progreso en los sistemas micro electro mecánicos (MEMS) y radio frecuencia RF.

A. Características

Estas redes se caracterizan principalmente por [1] [2]:

- Capacidad de auto organización.
- Tasas de transmisión de datos bajas.
- Cantidad de energía disponible para cada nodo limitada.
- Capacidad computacional y memoria de los nodos limitada.
- Baja potencia de transmisión.
- Topología de la red variable según su uso, o por falla de nodos.
- Capacidad de despliegue de gran cantidad de sensores y seguir funcionando a pesar del fallo de uno o más nodos.
- Capacidad de operar en ambientes hostiles.

B. Arquitectura de hardware

Cada dispositivo o nodo de la red está típicamente conformado por cuatro componentes [1]:

- Una o varias unidades de sensores: temperatura, humedad, etc.
- Una unidad de procesamiento o cómputo: un microcontrolador con memoria.
- Una unidad de comunicación: un *transceiver* de radio frecuencia.
- Una unidad de energía: baterías comunes, también paneles solares u otras formas de recolección de energía.

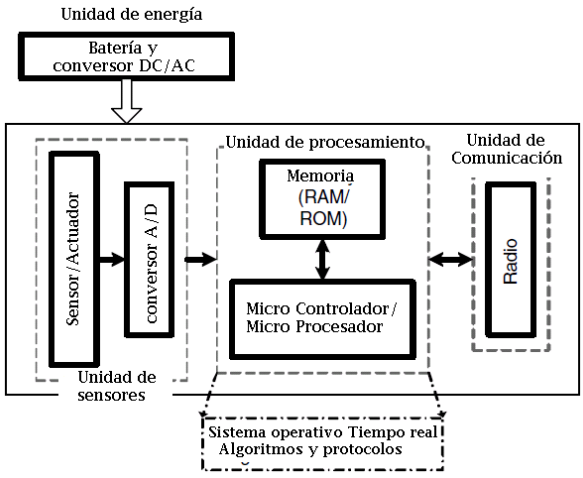

Fig 1. Arquitectura de hardware de un nodo sensor [1].

C. IEEE 802.15.4

Existen varias tecnologías o estándares para las redes inalámbricas de sensores, por ejemplo: WirelessHART, ZigBee, y 6lowpan, pero todas comparten al estándar IEEE 802.15.4 como base de operación.

Los dispositivos sensores utilizados en el presente proyecto también trabajan con estándar IEEE 802.15.4.

El estándar IEEE 802.15.4 especifica la capa física y la capa de control de acceso al medio (MAC) para las redes inalámbricas de área personal de baja velocidad (LR-WPANs) [3].

1) Componentes de la red WPAN IEEE 802.15.4

IEEE 802.15.4 define 2 tipos distintos de nodos [3]:

a) *FFD (Full Function Device):* son dispositivos de funcionalidad completa, un nodo FFD puede funcionar como coordinador PAN cuando se encarga de toda la red y no solo de su entorno, coordinador o como un nodo normal. Un FDD puede comunicarse con dispositivos RFD u otros FFD.

b) *RFD (Reduced Function Device):* dispositivos con funcionalidad reducida, son dispositivos sencillos con recursos y capacidad de memoria muy limitados. Un RFD sólo puede comunicarse con un FFD a la vez, y no pueden ser coordinadores.

2) Topología

Dependiendo de los requerimientos de la aplicación una red LR-WPAN IEEE 802.15.4 puede operar en 2 topologías: estrella y punto a punto (*peer to peer*) [3].

En cualquier topología, toda red necesita al menos un FFD que actúe como su coordinador. Los dispositivos se direccionan empleando 64-bits o un direccionamiento corto opcional de 16 bits.

3) Capa Física

La capa física puede transmitir en 3 bandas distintas no licenciadas:

- 868 MHz (868-868,8 MHz): Europa, permite un canal de comunicación (versión de 2003), extendido a tres en la revisión de 2006.
- 915 MHz (902-928 MHz): Norte América, hasta diez canales (2003) extendidos a treinta (2006).
- 2,4 GHz (2400-2483,5 MHz): uso en todo el mundo, hasta dieciséis canales (2003, 2006).

4) Capa MAC

Proporciona el acceso al canal físico de radio, permite transmitir tramas MAC a través del canal de radio reduciendo o evitando colisiones en el medio y es responsable de las siguientes tareas:

- Generar *beacons* de red si el dispositivo es un coordinador.
- Sincronización de la red con *beacons*.
- Soporta asociación y disociación de los nodos de la PAN.
- Funciones de seguridad (encriptación AES 128 bits).
- Utiliza CSMA-CA como mecanismo para acceso al canal.
- Opcionalmente garantiza slots de tiempo.
- Asegurar un enlace fiable con la capa MAC de nodos contiguos.

III. TinyOS y lenguaje NesC

Los sistemas operativos para los dispositivos de las redes inalámbricas de sensores son modestos, porque tienen que minimizar el código y deben estar optimizados para trabajar con pocos recursos de energía, memoria y procesamiento. Se han realizado muchos trabajos en sistemas operativos para nodos sensores como [2]: Nut/OS, Contiki, eCos, EYESOS, SenOS, MagnetOS, MANTIS, TinyOS, LiteOS, entre otros.

A pesar de la existencia de varios sistemas operativos, TinyOS hoy en día es el estándar de facto para las redes inalámbricas de sensores [2] y forma parte del presente proyecto.

A. TinyOS

TinyOS es un sistema operativo de código abierto diseñado para las redes inalámbricas de sensores, originalmente desarrollado como proyecto de investigación en la Universidad de California en Berkeley, pero actualmente tiene una comunidad internacional de desarrolladores y usuarios. Presenta una arquitectura basada en componentes que permite una rápida innovación e implementación mientras minimiza el

código para satisfacer las restricciones de memoria que son inherentes en las redes inalámbricas de sensores [4].

TinyOS básicamente es un conjunto de componentes que son incluidos a medida que son necesitados por las aplicaciones [5].

Actualmente el sistema operativo TinyOS tiene dos versiones. TinyOS 2.x es un rediseño de la versión de TinyOS 1.x, las aplicaciones desarrolladas en la versión 2.x no son compatibles con la versión 1.x, es decir que el código escrito en la versión 2.x no compilará en la versión 1.x. Sin embargo aplicaciones realizadas en TinyOS 1.x se pueden adaptar a la versión 2.x realizando ciertas modificaciones [6].

B. Características del diseño TinyOS

Para cumplir con los objetivos de diseño mencionados anteriormente, TinyOS se caracteriza por lo siguiente:

1) Arquitectura basada en componentes

Esta arquitectura permite la creación de aplicaciones ensamblando componentes que pueden ser componentes pre-existentes de librerías o pueden ser componentes creados por el programador. Permite un desarrollo de aplicaciones en forma modular, desde aplicaciones sencillas a complejas, ensamblando componentes. El lenguaje nesC proporciona la programación orientada a componentes.

2) Modelo de ejecución y concurrencia

En una típica aplicación de una red inalámbrica de sensores, un nodo es responsable de varios aspectos como sensado, procesamiento local de datos, transmisión de datos, enrutamiento, etc. Muchos de estos eventos requieren respuestas en tiempo real, por lo cual es necesario un manejo de concurrencia para reducir problemas respecto a recursos y restricciones en los nodos [7].

Si un fragmento de código sincrónico se ejecuta durante un largo tiempo, impide la ejecución del código de otros, y puede afectar negativamente a la capacidad de respuesta del sistema. Un componente debe ser capaz de dividir un cálculo grande en partes más pequeñas que puedan ejecutarse una a la vez. Además, hay ocasiones en que un componente necesita hacer algo, pero es preferible que su ejecución se lo haga más tarde.

A fin de cumplir lo mencionado anteriormente el modelo de ejecución y concurrencia de TinyOS consiste en tareas y eventos asincrónicos de hardware. Dando a TinyOS la posibilidad de aplazar el cómputo permitiendo ejecutar todo desde el primer proceso en espera.

Las tareas no son críticas en tiempo, están pensadas para hacer una cantidad mayor de procesamiento y se ejecutan por orden de llamada (FIFO). Los eventos son críticos en su ejecución y pensados para realizar un proceso pequeño (por ejemplo cuando un temporizador produce una interrupción o atender las interrupciones de un convertidor analógico-digital).

Si no existen tareas pendientes el procesador, se pasa a modo reposo para ahorro de energía hasta que un nuevo evento o tarea le haga despertar. Los eventos pueden interrumpir las tareas u otros eventos que se estén ejecutando, esto permite cumplir de mejor manera los requerimientos de tiempo real.

3) Modelo de comunicación

Un aspecto crítico en el diseño de TinyOS es el *networking*. Las comunicaciones de TinyOS se basan en el modelo *Active Messages* (AM) o Mensajes Activos que son pequeños paquetes de 36 bytes que incluyen un byte de *handler* ID [7]. El *handler* ID lo genera el emisor. Los paquetes AM también incluyen un campo destino, que se almacena en "Direcciones AM" para direccionar paquetes a nodos sensores en particular. La dirección AM de un nodo se la puede colocar en el momento de la instalación, utilizando los comandos *make install,n* o *make reinstall,n,* siendo *n* la dirección asignada.

Para la comunicación de radio no se recomienda el uso de TCP/IP debido a la restricción de memoria de los nodos y la carga de tráfico que generaría, la solución TinyOS planteada es el uso de los mensajes activos. El funcionamiento de la transmisión y recepción de paquetes AM es el siguiente: el emisor especifica el handler ID, solicita el envío y espera que llegue la respuesta de realizado. Cuando llega el mensaje el receptor se encarga de invocar el correspondiente manejador (*handler* ID), extrae el mensaje de la red lo procesa y envía un mensaje de respuesta con un evento asociado al *handler* ID. De esta forma no se generan bloqueos, esto es conocido también como operaciones divididas en fase (*split-phase*), la llamada a un servicio (comando) retorna inmediatamente y el correspondiente evento retorna cuando finaliza el proceso.

Los paquetes mensajes activos proporcionan una comunicación unificada para el canal de radio y el puerto serial (para nodos que funcionan como estación base) [7].

C. Estructura de directorios TinyOS

Se puede acceder al árbol de directorios TinyOS con el comando *cd $TOSROOT*. Dependiendo de la versión de TinyOS la variable de entorno *$TOSROOT* mostrará la ruta /opt/tinyos-1.x o /opt/tinyos-2.x.

La estructura de directorios TinyOS es como se muestra a continuación:

- apps/. Aplicaciones de ejemplo desarrolladas en nesC.
- doc/. Documentación y tutoriales.
- tools/. Herramientas externas a TinyOS para su uso conjunto con las aplicaciones nesC.
- tools/java/. Directorio base de los paquetes Java para TinyOS

- tos/. Directorio base de código fuente nesC del sistema operativo TinyOS.
- tos/interfaces/. Código fuente de todas las declaraciones de las interfaces TinyOS.
- tos/lib/. Librerías que extienden la funcionalidad al sistema, pero no se consideran esenciales por ejemplo TinySec.
- tos/platform/. Aquí se encuentran los *drivers* o código específico para las plataformas de hardware soportadas (por ejemplo telos, micaz, msp430)
- tos/system/. Todos los componentes principales que utiliza el sistema TinyOS (por ejemplo Main, TimerC, etc).
- tos/types/. Contiene archivos de cabecera .h usados por los componentes del sistema TinyOS.

D. Nesc (Networked embedded system C)

El sistema operativo TinyOS y sus aplicaciones están escritos en nesC, un nuevo lenguaje de programación orientado a componentes. NesC es una extensión del lenguaje C que soporta el diseño y modelo de ejecución de tinyos [8].

Los conceptos básicos que nesC ofrece son [8]:

- Separación entre la construcción y la composición. Las aplicaciones están formadas por un conjunto de componentes conectados entre sí.
- Hay dos tipos de componentes en nesC: módulos y configuraciones.
- Los componentes son conectados estáticamente a otros mediante sus interfaces. Esto mejora el tiempo de ejecución de la aplicación.
- Las interfaces son bidireccionales.
- Optimiza la generación de códigos, simplifica el desarrollo de aplicaciones, reduce el tamaño del código, y elimina fuentes potenciales de errores.
- El modelo de concurrencia de nesC está basado en la ejecución completa de tareas y eventos manejadores de interrupciones que pueden interrumpir tareas u otros eventos.

1) Componentes [8]

La unidad básica del código nesC es un componente. Un componente es un archivo. Los componentes se conectan mediante interfaces, estas conexiones se denominan "*wiring*". Las interfaces son el único punto de acceso al componente y son bidireccionales.

Los componentes utilizan interfaces de componentes ya existentes y proporcionan interfaces para poder ser utilizadas por otros componentes.

2) Interfaces [8]

Los componentes proporcionan y utilizan las interfaces. Las interfaces son una agrupación de funciones declaradas (comandos y eventos).

Las interfaces son bidireccionales (comando/evento): Un comando es una función implementada por el componente que proporciona la interfaz, un evento es una función implementada por el componente que utiliza la interfaz.

El lenguaje nesC utiliza flechas para determinar la relación entre interfaces. La dirección de una flecha (->) es siempre de un usuario a un proveedor. En otras palabras el componente que utiliza una interfaz está en el lado izquierdo, y el componente que proporciona el componente está en el lado derecho.

Por ejemplo, la expresión:

A.i -> B.i

Significa que el componente B proporciona la interfaz "i" y el componente A la utiliza. Si el proveedor está en el lado izquierdo, también puede utilizar una flecha hacia la izquierda:

B.i <- A.i

Sin embargo para facilitar la lectura, la mayoría de los cableados son de izquierda a derecha.

También existe el operador = y permite que dos interfaces nesC sean equivalentes.

3) Módulos y Configuraciones [8].

Existen dos tipos de componentes: módulos y configuraciones.

a) Módulos: son componentes que tienen variables y código ejecutable. Es decir proporcionan el código de la aplicación, implementando una o varias interfaces.

b) *Configuraciones:* se utilizan para ensamblar componentes unos con otros, esto es llamado el cableado o *wiring*. Cada aplicación nesC requerirá una configuración de alto nivel que conecte a los componentes utilizados, este archivo es esencial, ya que nesC utiliza este fichero para compilar y generar el ejecutable final que se cargará en el nodo, un error en este archivo no permitirá compilar la aplicación.

IV. DISEÑO E IMPLEMENTACIÓN DEL PROTOTIPO DE LA SOLUCIÓN DE MONITOREO WEB

El objetivo del desarrollo de esta aplicación es movilizar el ambiente de monitoreo de una red inalámbrica de sensores y proporcionar los detalles de los datos medidos de forma remota y amigable al usuario final. Para el uso de la aplicación no es necesario que el usuario final instale el entorno de desarrollo TinyOS en su equipo.

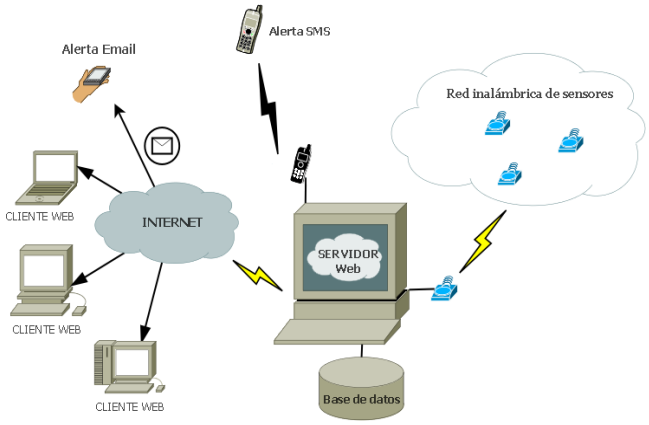

Fig 2. Bosquejo de la solución.

Fig 3. TRP2420(TelosB) con pack de baterías AA [9].

El servidor Web tiene conectado un nodo *gateway* en un puerto USB, todos los nodos sensores de la red inalámbrica envían sus datos al *gateway* el cual a su vez envía estos datos a una aplicación computacional que interpreta esta información y la almacena en una base de datos. Se puede acceder al servidor web desde el Internet o desde una Intranet según sea el caso, de esta manera un usuario puede monitorear los datos medidos por los nodos sensores remotamente desde una interfaz web. Solamente pueden utilizar la aplicación web los usuarios autenticados, la configuración del sistema únicamente la puede realizar el administrador. Se ha modificado la aplicación gráfica incluida en el sistema TinyOS llamada *oscilloscope* desarrollada en Java a fin que interprete los datos recolectados y los almacene en la base de datos.

Para este proyecto se han utilizado herramientas de código libre: el servidor web Apache, la base de datos MySQL, y el lenguaje PHP para la interfaz web. La aplicación web notificará las alarmas que se generen de acuerdo a la configuración establecida por el administrador, y permitirá que un usuario autenticado pueda consultar los datos medidos y graficarlos en tiempo real o por un rango de tiempo definido. Las alertas se notifican automáticamente con mensajes de texto mediante un celular, o por email mediante un servidor de correo.

A. Hardware empleado

Los componentes que conforman parte del prototipo son los siguientes:

1) Dispositivos TelosB

Los sensores Telos Revisión B, conforman el prototipo de la red inalámbrica de sensores, y sus valores medidos se almacenan en la base de datos del servidor.

Se trabaja con los dispositivos TPR2420 y TPR2400. El nodo TPR2420 es igual en funcionalidad que el TPR2400, excepto que incluye sensores ambientales pre-instalados.

a) Características principales:

- La comunicación inalámbrica cumple con el estándar IEEE 802.15.4.
- Frecuencia de operación 2.4 a 2.4835 GHz, banda ISM (*Industrial Scientific and Medical*).
- Velocidad 250 kbps.
- Antena integrada con rango de 20-30 m en interiores y 75-100 m en exteriores.
- Bajo consumo de energía, gracias al uso del microcontrolador Texas Instruments MSP430 con 10kB de RAM.
- Recolector de datos, programación y alimentación vía USB.
- Posibilidad de integrar sensores para luz, temperatura y humedad.
- Trabaja con el sistema operativo de código abierto TinyOS 1.1.11 o superior.

2) Equipo servidor

Para un buen desempeño del sistema se recomienda un servidor con las siguientes características:

- 1024 MB de memoria RAM
- 80 GB de disco duro.
- 2 GHz en procesador.
- Servidor Web con soporte para PHP 5 y SSL.
- Base de datos MySQL 5
- Puertos USB 2.0
- Tarjeta de red Ethernet 10/100 Mbps

3) MODEM GSM

De los diferentes teléfonos celulares existentes en el mercado, se trabaja con el económico equipo Motorola C385 debido a que el modem GSM incorporado cumple con los requisitos para envíos de SMS desde el computador. Este teléfono puede

ser reemplazado con cualquier otro teléfono que disponga de un modem GSM que soporte comandos AT.

B. Software empleado

El producto fue desarrollado bajo el sistema operativo Windows, concretamente la versión XP Profesional. Sobre este sistema se monta un servidor Apache versión 2.2, se incluye como Sistema Gestor de Bases de Datos MySQL versión 5.1, y se lo maneja con la aplicación Web denominada PHP MyAdmin.

Se emplea como lenguaje de implementación a PHP versión 5.2.9 junto con HTML para el desarrollo de la interfaz Web y CSS para proporcionarle estilos a las páginas, la librería JpGraph 3.0.7 y java script para mostrar los gráficos en tiempo real, para el envío de los mensajes de texto se utiliza Java. Con todos estos requisitos, se ejecutará el sistema de monitoreo web.

C. Prototipo de la red inalámbrica de sensores

Los nodos sensores utilizados son los fabricados por la empresa *Crossbow Technology*, para fines prácticos el prototipo de red inalámbrica de sensores implementado consiste de tres dispositivos TelosB (un nodo *gateway* y dos nodos sensores). Ya que el nodo *gateway* necesitará mayor energía que el resto de nodos, permanecerá conectado al puerto USB del computador, para cumplir con este objetivo resulta conveniente implementar una topología en estrella para el prototipo de la red.

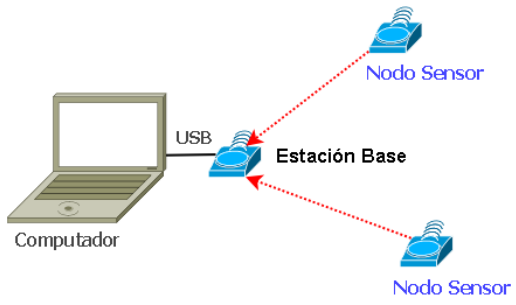

Fig 4. Topología de red del prototipo.

D. Metodología de desarrollo

El proceso de desarrollo del sistema de monitoreo Web está basado en el estándar UML o lenguaje unificado de modelado, que permite la construcción de sistemas por medio de conceptos orientados a objetos [10]. UML es independiente de la metodología, por este motivo es importante escoger un proceso de desarrollo adecuado para elaborar el sistema. Se ha elegido como metodología al Proceso Unificado porque utiliza al lenguaje UML como parte fundamental para modelar, construir y documentar los elementos que forman parte de un sistema.

1) Arquitectura del sistema de monitoreo

En el diseño actual de sistemas informáticos se suele usar las arquitecturas multinivel o programación por capas. Se desarrolla el sistema de software en tres capas para evitar tener todo el código mezclado y resulte más cómodo el desarrollo y su mantenimiento.

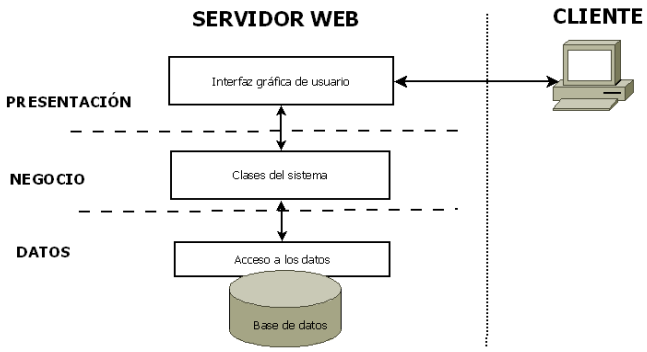

Fig 5. Arquitectura del sistema de monitoreo.

2) Diseño de la interfaz del sistema

Se establece un diseño de interfaz general para las páginas Web del sistema. La interfaz Web tiene una estructura básica con un encabezado, un menú, el contenido y el pie de página. En el contenido se muestran formularios y datos del monitoreo solicitados por el usuario. Las opciones del menú cambian dependiendo si el usuario es un operador o administrador.

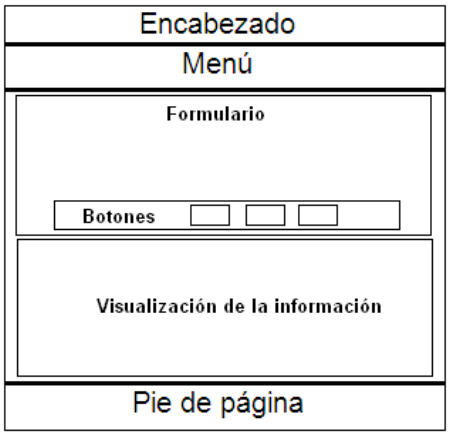

Fig 6. Estructura de las páginas web del sistema.

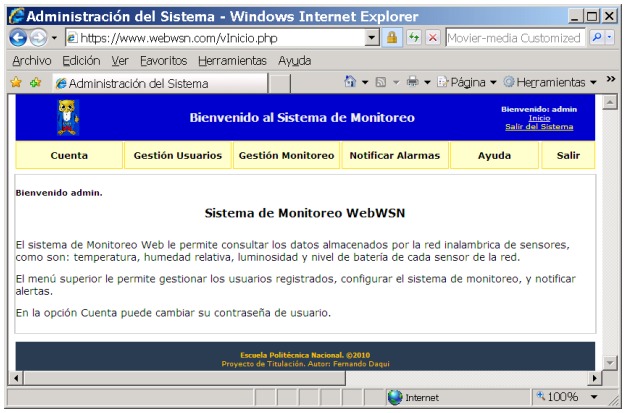

Fig 7. Ejemplo de las páginas Web del sistema.

3) Diseño de seguridad del sistema

- El sitio estará protegido con autenticación mediante un nombre de usuario y contraseña.
- Para cada usuario autenticado el sistema desplegará ciertos módulos de acuerdo al perfil que posea.
- Las contraseñas del sistema se almacenan codificadas en la base de datos utilizando un algoritmo de encriptación.

E. Puesta en marcha del sistema

Una vez que se tiene instalado el sistema operativo TinyOS y programados los nodos de la red, mediante el comando *motelist* se determina el puerto COM del nodo estación base, se inicia la aplicación *SerialForwarder* ejecutando:

$ java net.tinyos.sf.SerialForwarder –comm serial@COM4:telos

Para iniciar la aplicación gráfica modificada se ejecuta el siguiente comando:

$ java net.tinyos.oscope.oscilloscope

Esta aplicación se conecta al SerialForwarder para mostrar gráficamente los datos de los nodos sensores de la red, cada vez que recibe un paquete la consola muestra un mensaje de conexión a la base de datos como se observa en la figura 8.

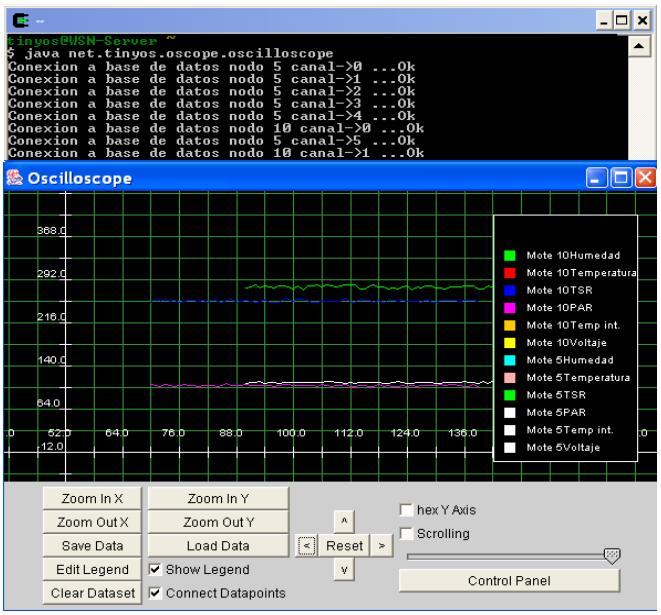

Fig 8 Aplicación Oscilloscope GUI almacenando mediciones a la base de datos MySQL.

Antes de iniciar el sistema Web, se deben arrancar los servidores Apache y MySQL, para las alertas por email correo, se debe configurar y levantar el servidor de correo que sea elegido, y para las alertas de mensajes de texto, el celular GSM debe conectarse al puerto USB del computador e instalado el respectivo modem GSM.

La primera página muestra un cuadro para ingresar el *login* y *password* asignado al usuario.

Fig 9. Página principal del sistema.

El ingreso como administrador permite gestionar las cuentas de usuarios que utilizan el sistema, gestión de monitoreo para configurar el sistema de monitoreo y notificar alarmas.

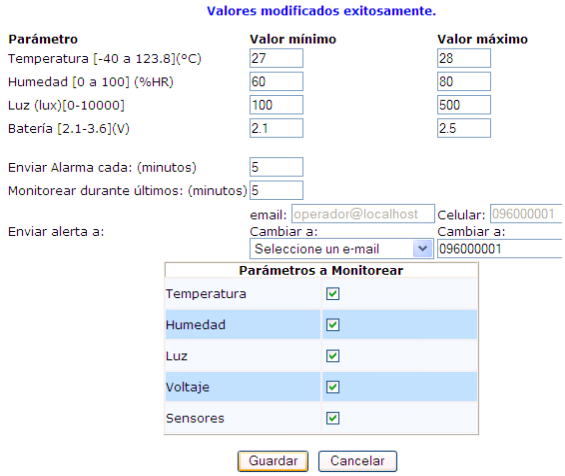

Fig 10 Configuración del sistema de monitoreo.

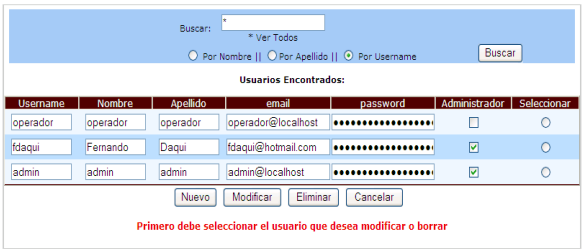

Fig 11. Configuración de usuarios del sistema.

El ingreso como operador permite la consulta de datos almacenados, alarmas generadas, gráficas en tiempo real y por rango de tiempo definido, el monitoreo de los valores promedio medidos de la red en tiempo real e individualmente de cada nodo mediante consulta.

Fig 12. Monitoreo de la red en tiempo real.

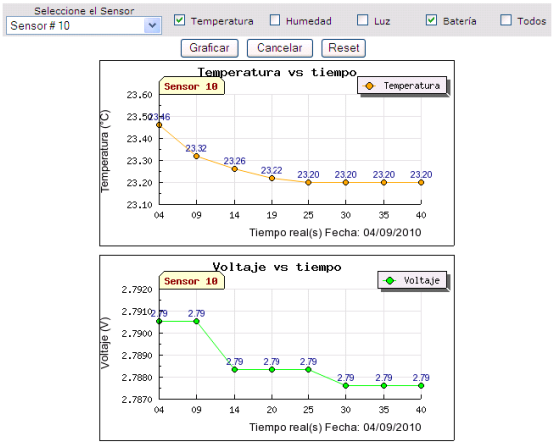

Fig 13. Gráfico en tiempo real

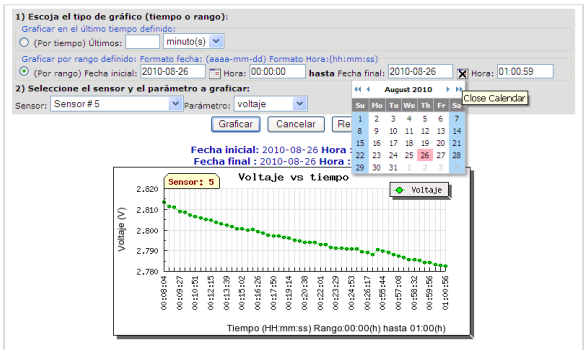

Fig 14. Gráfico por rango de tiempo.

Fig 15. Consulta de alarmas generadas

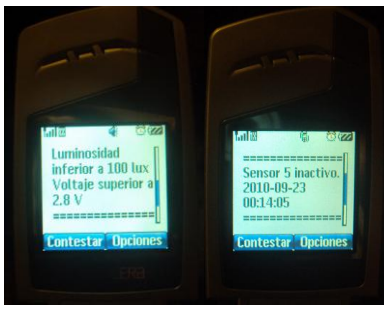

Fig 16. Mensajes de texto enviados al celular seleccionado.

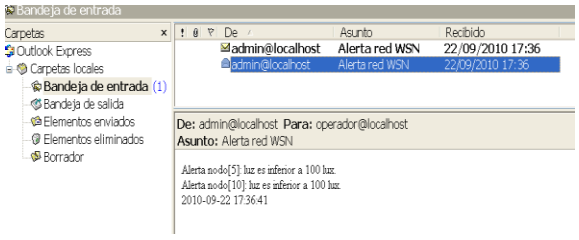

Fig 17. Alertas enviadas al email seleccionado.

V. CONCLUSIONES

- El sistema desarrollado se puede adaptar a varias aplicaciones, por ejemplo medir variables de temperatura, humedad, luz, en *data centers*, museos, hábitat de animales y vegetación, invernaderos, edificios, etc.; facilitando de este modo al usuario final el monitoreo de la red utilizando solamente un navegador Web.

- Hoy en día TinyOS se define como el estándar de facto para las redes inalámbricas de sensores principalmente por su amplio uso por parte de desarrolladores, gran cantidad de grupos de trabajo, amplia documentación de soporte y las múltiples aplicaciones desarrolladas. Estas características convierten a TinyOS en la alternativa más segura y conveniente a la hora de desarrollar una red inalámbrica de sensores.

- Las redes inalámbricas de sensores son una gran alternativa ecológica por su bajo consumo de energía, su baja potencia de emisión de radiación y alta durabilidad, además es posible utilizar paneles solares en lugar de baterías ocasionando el mínimo o casi ningún impacto al medio ambiente y seres vivos, otro punto a favor que hará muy popular a esta tecnología en un futuro cercano.

- El lenguaje nesC y el sistema operativo TinyOS se encuentran profundamente relacionados, cualquier avance o desarrollo de uno se verá reflejado en el otro.

- TinyOS al ser un sistema de código abierto permite implementar su propia capa de comunicación sobre IEEE 802.15.4 como son los mensajes activos, lo cual es una gran ventaja al no depender de estándares propietarios para capas superiores.

- Si se requiere mayor eficiencia en el envío de alertas automáticas, se recomienda implementar el envío de mensajes de texto en una aplicación independiente y no llamarla directamente desde PHP. Pero antes de implementar esta nueva aplicación se debe considerar que los teléfonos celulares modernos ya reciben emails y para fines prácticos se puede prescindir de los mensajes de texto.

VI. REFERENCIAS

[1] Mohammad Ilyas, Imad Mahgoub, Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems, CRC Press LLC, 2005.

[2] Kazem Sohraby, Daniel Minoli, Taieb Znati, Wireless Sensor Networks Technology - Protocols And Applications, WILEY, 2007.

[3] IEEE 802.15,"Standards.ieee.org", standards.ieeeorg, http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf

[4] TINYOS COMMUNITY FORUM, "TinyOS Documentation Wiki", TinyOS.net http://docs.tinyos.net/index.php/Main_Page

[5] Gay David, Levis Phil, Culler David, Software Design Patterns for TinyOS, LCTES'05, Chicago, 2005.

[6] TINYOS COMMUNITY FORUM, "TinyOS 2.0 Overview", tinyos.net, http://www.tinyos.net/tinyos-2.x/doc/html/overview.html

[7] Levis Philip, Madden Sam, Polastre Joseph, Szewczyk1 Robert, Kamin Whitehouse, Gay David, Hill J, Welsh M, Brewer E, Culler David, "TinyOS: An Operating System for Sensor Networks",2005

[8] Gay David, Levis Phil, Culler David, Brewer Eric, "nesC 1.1 Language Reference Manual", Chicago, Mayo 2003, http://nescc.sourceforge.net/papers/nesc-ref.pdf

[9] CROSSBOW, "TPR2400/2420 Quick Start Guide", Crossbow Technology, http://www.xbow.jp/telos_guide_7430-0380-01_a.pdf

[10] JACOBSON I., BOOCH G., RUMBAUGH J., El Proceso Unificado de Desarrollo de Software, Pearson Educación, Madrid-España, 2000.

[11] PHP, "PHP: Hypertext Preprocessor", http://www.php.net/

VII. BIOGRAFÍAS

Oswaldo Fernando Daqui Solano, nació en Quito-Ecuador, el 23 de mayo de 1980. Realizó sus estudios secundarios en el colegio municipal Sebastián de Benalcázar. Ha trabajado en varios proyectos tecnológicos en IBM del Ecuador. Se graduó como Ingeniero en Electrónica y Redes de Información en el 2011. Actualmente trabaja en el área de redes y comunicaciones de la Dirección Metropolitana de Informática.
Areas de interés: informática y redes, switches, routers, seguridad.
(fernando.daqui@gmail.com)

Fabio M. González G. Nació en Guaranda, Ecuador, el 8 de abril de 1966. Obtuvo su título de Ingeniero en Electrónica y Telecomunicaciones en la Escuela Politécnica Nacional, en 1991. Realizó estudios de especialización en Transmisión Digital y Redes de Computadoras en Quebec, Canadá, en 1995. Realizó una especialización en Redes Inalámbricas de Sensores en la Universidad de la Suiza Italiana, Lugano, Suiza, en el 2008. Actualmente se desempeña como Profesor Principal a Tiempo Completo y Coordinador de la Carrera de Electrónica y Redes de Información de la Escuela Politécnica Nacional.
(fabio.gonzalez@epn.edu.ec)