Revista EIA, ISSN 1794-1237 Número 17, p. 151-166. Julio 2012
Escuela de Ingeniería de Antioquia, Medellín (Colombia)

EVALUACIÓN DEL DESEMPEÑO EN REDES INALÁMBRICAS DE SENSORES MEJORADAS CON AGENTES MÓVILES

Alcides Montoya*
Demetrio Arturo Ovalle**

RESUMEN

La reconfiguración, reprogramación y despliegue de nuevas tareas computacionales en redes inalámbricas de sensores es un problema no resuelto satisfactoriamente en la actualidad. Este artículo propone la evaluación del desempeño en redes inalámbricas de sensores mejoradas con agentes móviles inteligentes como mecanismo de reprogramación autónoma. El método utilizado para la evaluación del desempeño se fundamenta en la medida del consumo de energía durante el proceso de migración de los agentes móviles inteligentes entre los nodos sensores y en el cálculo del tiempo de convergencia de la red, definido como el tiempo que tarda la red en pasar de un estado a otro; en los experimentos se refiere al retardo durante el cambio del tiempo de muestreo para toda la red. La solución más eficiente, que fue probada y evaluada en una red inalámbrica formada por 40 nodos que detectan fugas de amoniaco en tiempo real, determinó que el punto clave consiste en disminuir el consumo de energía producto de las confirmaciones y retransmisiones innecesarias de datos y procedimientos, desde los nodos sensores hasta la estación base. Este hecho representa, además de la disminución en el consumo energético, un ahorro significativo en el tiempo de convergencia de la red.

PALABRAS CLAVE: red inalámbrica de sensores; agentes móviles inteligentes; evaluación del desempeño; reprogramación autónoma.

* Físico, Universidad de Antioquia; Magíster en Informática, Universidad EAFIT; Doctor (c) en Ingeniería de Sistemas y Profesor Asistente, Universidad Nacional de Colombia, Sede Medellín. Miembro del Grupo de Instrumentación Científica e Industrial (GICEI) y del Grupo de Investigación y Desarrollo en Inteligencia Artificial (GIDIA). Medellín, Colombia. amontoya@unal.edu.co

** Ingeniero de Sistemas y Computación, Universidad de los Andes, Bogotá; D.E.A. en Informatique, Institut National Polytechnique de Grenoble, Francia; Docteur en Informatique, Université Joseph Fourier, Francia. Profesor Titular y Director del Grupo de Investigación y Desarrollo en Inteligencia Artificial (GIDIA), Escuela de Sistemas, Universidad Nacional de Colombia, Sede Medellín. Medellín, Colombia. dovalle@unal.edu.co

Artículo recibido 1-VII-2011. Aprobado 13-VI-2012
Discusión abierta hasta diciembre de 2012

PERFORMANCE EVALUATION OF WIRELESS SENSOR NETWORKS IMPROVED WITH MOBILE AGENTS

ABSTRACT

The reconfiguration, reprogramming, and deployment of new computational tasks in wireless sensor networks are complex and represent a problem satisfactorily unresolved at present. The aim of this paper is to propose the performance evaluation of the use of mobile intelligent agents as autonomous rescheduling mechanism in such networks. The method used for performance evaluation is done by measuring the energy consumption in the migration of mobile intelligent agents among the sensor nodes of the system and calculating the convergence time of the network, defined as the time it takes for the network to move from one state to another; in experiments it refers to the delay in changing the sampling time for the entire network. The most efficient solution, which was tested and evaluated in a network is composed of 40 nodes that detect in real time ammonia leaks, determined that the key issue is to reduce the expenditure of unnecessary energy in transmission from the wireless sensors to the base station, while avoiding unnecessary confirmations and transmissions of data and procedures among sensor nodes. This fact represents besides the reduction in the network energy consumption, a very significant saving for convergence time of the network.

KEY WORDS: wireless sensor networks; intelligent mobile agents; performance evaluation; autonomous reprogramming.

AVALIAÇÃO DO DESEMPENHO EN REDES DE SENSORES SEM FIO MELHORADAS COM AGENTES MÓVEIS

RESUMO

A reconfiguração, reprogramação e implantação de novas tarefas computacionais em redes de sensores sem fio é um problema não resolvido de modo satisfatório na atualidade. Este artigo propõe a avaliação do desempenho em redes de sensores sem fio melhoradas com agentes móveis inteligentes como mecanismo de reprogramação autônoma. O método utilizado para a avaliação do desempenho fundamenta-se na medida do consumo de energia durante o processo de migração dos agentes móveis inteligentes entre os nós sensores e no cálculo do tempo de convergência da rede, definido como o tempo que demora a rede de passar de um estado a outro; nos experimentos se refere ao retardo durante a mudança do tempo de amostragem para toda a rede. A solução mais eficiente, que foi provada e avaliada em uma rede sem fio formada por 40 nós que detectam fugas de amoníaco em tempo real, determinou que o ponto-chave consiste em diminuir o consumo de energia, produto das confirmações e retransmissões desnecessárias de dados e procedimentos, desde os nós sensores até a estação base. Este fato representa, além da diminuição no consumo energético, uma poupança significativa no tempo de convergência da rede.

PALAVRAS-CÓDIGO: rede de sensores sem fio; agentes móveis inteligentes; avaliação do desempenho; reprogramação autônoma.

1. INTRODUCCIÓN

Una red inalámbrica de sensores (RIS) es un sistema formado por decenas o cientos de pequeñas estaciones denominadas nodos sensores. Los nodos están compuestos a su vez por un grupo especializado de sensores y transductores que poseen una infraestructura de comunicación necesaria para almacenar datos, notificar alguna condición específica y hacer medición y seguimiento de variables físicas en los entornos donde son desplegados. Utilizando una RIS se pueden medir las siguientes variables: temperatura, humedad, presión, dirección y velocidad del viento, intensidad de iluminación, vibración, intensidad del sonido, voltajes en líneas de potencia, energía activa en redes eléctricas, concentraciones

químicas, niveles de contaminación, funciones vitales del cuerpo, concentraciones de gases, entre otras (Ovalle, Restrepo y Montoya, 2010). Los nodos sensores tienen muy pocas capacidades de cómputo por sí solos, pero cuando trabajan en equipo, su capacidad de procesamiento y su área de cubrimiento se hacen óptimas (Culler, Estrin y Srivastava, 2004).

Los nodos de una RIS son autónomos, y al ser conectados, constituyen un sistema distribuido de cómputo que trabaja de forma cooperativa para medir variables físicas y cambios en condiciones ambientales. Cada nodo está equipado con un radio, un pequeño microcontrolador, una fuente de energía que usualmente es una batería (Römer y Mattern, 2004). Las RIS están sujetas a restricciones más estrictas que otros dispositivos electrónicos afines, como son los teléfonos móviles o los computadores portátiles. La red completa suele estar bajo la administración de un elemento controlador denominado estación base (EB), cuya función principal es actuar como pasarela para otras redes, almacenar datos y conformar la red como tal. Es importante señalar que todos los paquetes de datos que provienen de los nodos son enviados hacia la EB.

En otras palabras, una RIS suele tener dispositivos de funcionalidad completa (DFC), que son las estaciones base o pasarelas y dispositivos de funcionalidad reducida (DFR) que toman los datos y los envían a los DFC. Los DFC tienen energía disponible todo el tiempo, mayor capacidad de cómputo y posibilidades de almacenamiento. Además, los DFR deben funcionar con pilas y deben dormir grandes períodos, con el fin de optimizar el uso de energía (Akyildiz *et al.,* 2002).

Las RIS pueden clasificarse en cinco tipos, dependiendo de las condiciones de trabajo, la dificultad de implementación y la aplicación específica, así se tienen: RIS sobre tierra (Akyildiz *et al.,* 2002; Toumpis y Tassiulas, 2006; Yick, Mukherjee y Ghosal, 2008), RIS bajo tierra (Li y Liu, 2007), RIS bajo el agua (Akyildiz, Pompili y Melodia, 2004), RIS multimedia (Akyildiz, Melodia y Chowdhury, 2007) y RIS móviles.

2. LIMITACIONES DE LAS RIS

Existen diversas limitaciones en una RIS, relacionadas con el consumo de energía, el desarrollo rápido de aplicaciones y la reprogramación de nuevas tareas computacionales (Ovalle, Montoya y Muñoz, 2011). A continuación se describe cada una de ellas.

La primera y principal limitación de una RIS es la baja disponibilidad de energía en los nodos. Su componente DFR funciona por lo general con dos pilas AA o similares a las usadas por los teléfonos móviles que como máximo pueden garantizar 720 mA, las cuales, en el mejor de los casos tienen una duración de meses. Es imposible para este tipo de redes considerar cambios permanentes de las pilas y desafortunadamente, en un futuro cercano, no se visualiza una optimización en ellas, de forma que puedan los nodos funcionar por mucho tiempo. Al respecto se han propuesto diversas soluciones, que van desde el uso de hardware que funciona con un mínimo de corriente, hasta protocolos que optimizan el proceso de inactividad de los nodos sensores.

La segunda gran limitación de las RIS consiste en el desarrollo de aplicaciones que luego puedan ser reprogramadas; en la actualidad, una vez que la RIS ha sido programada y desplegada, se hace prácticamente imposible usarla para otras tareas o cambiar sus parámetros de configuración iniciales. Se han propuesto diversas soluciones a este problema de reprogramación y de cambios en nuevas tareas para las RIS, sin embargo, no se posee un método estándar que permita la reprogramación, reconfiguración y asignación de nuevas tareas computacionales.

La tercera y última limitación está relacionada con el desarrollo rápido y ágil de aplicaciones para las RIS. Se ha encontrado en esta parte el principal escollo para que se adopte la tecnología RIS a gran escala en el mundo. Es importante resaltar que solo se cuenta con pocos desarrollos importantes e incluso "construidos a la medida" para el despliegue rápido de aplicaciones.

3. REPROGRAMACIÓN Y RECONFIGURACIÓN DE RIS

La tarea de reprogramación y reconfiguración de una RIS se define como la capacidad de desplegar nuevas aplicaciones computacionales de forma dinámica, sin la necesidad de intervención humana en la red (o sea, reprogramar de forma manual cada uno de los nodos y mediante un computador conectado a él). En general, los desarrolladores pueden estar interesados en hacer mejoras a las aplicaciones previamente desarrolladas, o pretenden construir una nueva aplicación o quizás cambiar alguno de los parámetros de los nodos (v. gr., el tiempo de muestreo o el tiempo de reposo del nodo). Así mismo, podría ser de interés reconvertir por completo la aplicación o solucionar problemas de errores en el software. Por otro lado, se puede necesitar la red para realizar una nueva tarea de muestreo o cuando los administradores de la red desean implementar actualizaciones del software. En conclusión, la reprogramación y la reconfiguración se hacen necesarias para que las RIS puedan adaptarse a los cambios en el entorno.

4. AGENTES MÓVILES

En la última década, proveniente de la IAD (inteligencia artificial distribuida) y de los SMA (sistemas multiagente) (Sycara, 1998; Wooldridge, 2009), ha surgido un nuevo modelo de programación basado en agentes móviles inteligentes (Tynan *et al.*, 2005), que son procesos capaces de moverse a través de una red informática (Zhou y Gao, 2010), ya sea una red de área local o una red de área amplia, migrando o clonando su código y estado, de una máquina a otra. De esta forma, los agentes móviles interactúan con dispositivos externos, procesan y recopilan información, para luego volver a su origen con los datos obtenidos. Los agentes móviles han tenido gran aceptación gracias a su característica de movilidad, ya que se ha probado que es mucho más eficiente que un agente se traslade hasta una ubicación remota, haga una búsqueda, filtre, procese la información y regrese con los resultados al punto origen, que, por el contrario, migrar la información sin realizar ningún tipo de procesamiento, lo cual implica costos de comunicación, consumo de memoria, procesamiento local, entre otros (Russell y Norving, 2003). La utilización de agentes móviles en RIS debe ser realizada con precaución debido a que las confirmaciones y retransmisiones de datos y procedimientos en la RIS pueden acarrear un alto consumo energético, lo cual representa un deterioro en el tiempo de vida de la red.

Algunos ejemplos de arquitecturas de programación que se han construido permitiendo utilizar agentes móviles en RIS son los siguientes: Impala (Liu y Martonosi, 2003), Agilla (Fok, Roman y Lu, 2009a) y ActorNet (Cheong, 2007). Con estas es posible solventar problemas tales como comunicación, procesamiento y almacenamiento de forma transparente entre el programador y el hardware del dispositivo. Los agentes son ligeros y cuentan con una pequeña sobrecarga de transmisión y consumo de energía.

5. TRABAJOS RELACIONADOS

La reprogramación de nodos sensores en una RIS comprende dos fases: la primera es la diseminación del código en las redes; este paso involucra el envío del código de forma eficiente y su recepción por parte del nodo destino. La segunda corresponde a la ejecución del código. En la primera fase, tal vez la más crítica, se hace necesario el desarrollo de protocolos de diseminación eficientes en consumo energético. Se han propuesto en la literatura diferentes métodos para abordar ambas fases. Existen métodos basados en marcos de desarrollo de software específicos (middleware), en sistemas operativos, en scripts, en bases de datos y en máquinas virtuales. Las pruebas de reprogramación suelen hacerse en redes habilitadas para uno o múltiples saltos. De esta forma, se calcula en cada propuesta adoptada el tiempo gastado en la reprogramación y su eficiencia en el consumo energético.

Dentro de las propuestas desarrolladas que utilizan algunos de los métodos enunciados están las siguientes:

Maté. Es una propuesta basada en una máquina virtual que se ejecuta con el sistema operativo TinyOS (TinyOS, 2011). Provee una interfaz de programación de alto nivel, permitiendo el desarrollo de los programas, relativamente simples, al mismo tiempo que reduce el tamaño de la aplicación a un rango de 100 bytes. La distribución del código se realiza mediante una difusión viral de paquetes pequeños –cuya programación es del tipo bytecode–, cada uno de los cuales contiene 24 instrucciones y es capaz de autorreplicarse (Levis y Culler, 2002). Aunque en esta propuesta hay esfuerzos importantes en disminuir el consumo de memoria, los autores, sin embargo, no hacen uso de técnicas de inteligencia artificial para disminuir el consumo energético de los nodos sensores.

Trigger. Es un modelo de programación para sistemas basados en eventos y nodos heterogéneos caracterizados por no utilizar agentes inteligentes ni programación distribuida. Cada nodo posee un conjunto de tres tablas indexadas (expresiones y variables por referencia), las cuales se modifican de manera centralizada desde la estación base, durante el proceso de reprogramación. Es importante señalar que esta propuesta opera de forma independiente del esquema de enrutamiento usado por la RIS y funciona de modo similar a Maté. El modelo de programación Trigger aprovecha la característica de que en una RIS es más frecuente la reconfiguración de parámetros que la reprogramación completa de un nodo (Tompkins *et al.,* 2011).

XNP. Es una propuesta de reprogramación que disemina código nada más dentro del radio de comunicación de la estación base. Se considera un método de un solo salto y está diseñado sólo para nodos de tipo Berkeley TinyOS/MOTE (Crossbow, 2003; MEMSIC, 2011). Debido a sus limitaciones en hardware, estos nodos no soportan la utilización de agentes inteligentes.

Deluge. Emplea un enfoque de reprogramación de múltiples saltos, donde los nodos intermedios ayudan en la reprogramación. De esta forma, un programa completo que es enviado a un nodo, se divide en múltiples segmentos, cada uno de los cuales contiene un número fijo de paquetes que se distribuyen a través de la RIS (Hui y Culler, 2004). Es importante señalar que esta propuesta no está diseñada para usar mecanismos de inteligencia artificial.

SensorWare. Permite mover código escrito en lenguaje de programación TCL (Tool Command Language) de un nodo a otro; provee el soporte para múltiples aplicaciones que se ejecutan de forma concurrente en la misma red. Cuando un código migra de un nodo a otro, se debe calcular la energía requerida para el código especificado; este parámetro sirve para definir si el nodo destino acepta o no el código (Boulis y Srivastava, 2002). Este enfoque no hace uso de agentes inteligentes para disminuir el consumo energético, debido a que su concepción y arquitectura no lo permiten.

MASPOT. Es quizás la propuesta más relacionada con la nuestra, en la medida en que usan agentes móviles inteligentes (Lopes, Assis y Montez, 2011). Los agentes se basan en Java y se utilizan en la reprogramación de la plataforma Sun SPOT para realizar migración de código y cambios pequeños dentro de los programas. La versión de Java en la cual se ejecutan los agentes móviles se denomina Squawk (Simon *et al.,* 2005). La evaluación que hacen los autores del costo en consumo energético de la migración y del uso de memoria en los nodos Sun SPOT, comparada con las otras técnicas, resulta muy adecuada. En efecto, comparado este método con los anteriores, se obtiene que el costo en consumo de energía es de solo el 0,03 % y el uso de memoria es apenas el 1,5 % de la memoria flash disponible. Ello hace muy prometedor este enfoque, pero sólo es aplicable a la plataforma Sun SPOT.

MAWSN (mobile agent based wireless sensor network). Es una arquitectura propuesta por Chen *et al.* (2006) basada en agentes móviles integrados en

RIS. La simulación realizada por ellos muestra que el modelo MAWSN exhibe mejor rendimiento que el modelo clásico de las RIS basado en la arquitectura cliente/servidor, en términos de consumo de energía en el proceso de transmisión y recepción de paquetes de datos, sin embargo, el modelo MAWSN presenta un alto retardo entre los nodos bajo ciertas condiciones y parámetros, como el número de nodos, tamaño del código procesado, tamaño de los datos obtenidos por cada sensor, entre otros.

6. MODELO MATEMÁTICO PARA CALCULAR EL CONSUMO DE ENERGÍA EN LOS NODOS DE LA RIS

Los nodos en una RIS están formados por diferentes módulos, a saber: microprocesador, radio de emisión/transmisión, sensores y fuente de energía. Para cada uno de estos componentes se propone un modelo matemático que permite después controlar los experimentos y comparar las simulaciones con los resultados reales.

6.1 Modelo de energía del microprocesador en la RIS

El microprocesador soporta tres estados de operación: dormido, disponible y ejecutando una tarea. Así mismo, posee cinco estados de transición entre cada uno de los nodos (Lu, Xu y Ying, 2005). La energía del microprocesador puede calcularse como:

$$E_{cpu} = E_{cpu-state} + E_{cpu-change}$$
$$E_{cpu} = \sum_{i=1}^{m} P_{cpu-state}(i)\, T_{cpu-state}(i) + \sum_{j=1}^{n} N_{cpu-change}(j)\, e_{cpu-change}(j) \quad (1)$$

Donde $P_{cpu-state}(i)$ es la potencia del estado i que puede consultarse en el manual de referencia del microprocesador; $T_{cpu-state}(i)$ es el intervalo de tiempo en el estado i, el cual es un valor estadístico que puede ser encontrado con el modelo propuesto; $N_{cpu-change}(j)$ corresponde a la frecuencia de transición de estado j y $e_{cpu-change}(j)$ es el consumo de energía unitario en el estado de transición j el cual puede expresarse como:

$$e_{cpu-change}(j) = T_{init-end}(j)\left(\frac{P_{init}(j) + P_{end}(j)}{2}\right) \quad (2)$$

Donde $P_{init}(j)$ y $P_{end}(j)$ corresponden a la potencia en el estado $init$ y end respectivamente, durante el estado de transición j; $T_{init-end}(j)$ es el intervalo de tiempo del estado de transición j desde el estado $init$ al estado final end.

6.2 Modelo de energía para el radio de comunicación

El módulo de comunicación de cada nodo incluye un radio banda-base y un sistema de radiofrecuencia. El radio normalmente tiene seis estados (T_x, R_x, Off, Idle, Sleep, CCA/ED) y nueve estados de transición (Wang, Xiang y Hu, 2009).

La función de consumo energético del radio puede expresarse como:

$$E_{trans-state} = E_{TX} + E_{RX} + E_{Idle} + E_{sleep} + E_{CCA}$$
$$E_{trans-state} = \sum_{i=1}^{N_{TX}} P_{TX}\frac{L_i}{R} + \sum_{i=1}^{N_{RX}} P_{RX}\frac{L_i}{R} + P_{Idle}T_{Idle} + P_{sleep}T_{sleep} + P_{CCA}T_{CCA} \quad (3)$$
$$E_{trans-state} = \sum_{i=1}^{N_{TX}} V_{tr}I_{TX}\frac{L_i}{R} + \sum_{i=1}^{N_{TX}} V_{tr}I_{RX}\frac{L_i}{R} + V_{tr}(I_{Idle}T_{Idle} + I_{sleep}T_{sleep} + I_{CCA}T_{CCA})$$

Donde E_x, P_x, I_x y T_x son respectivamente la energía consumida, la potencia y la corriente eléctrica y el intervalo de tiempo del radio en el estado x; V_{tr} es el voltaje normal de trabajo del radio; L_i es la longitud del i° paquete enviado o recibido; R es la cantidad de datos transmitidos; N_{Tx} y N_{Rx} se definen como el número de paquetes enviados y recibidos. $E_{trans-change}$ será expresada en la ecuación (4) para el radio. La variable $j = 1,2,..n$ es el tipo de transición ($n=9$); $N_{trans-change}(j)$ corresponde a la frecuencia de transición al estado j, y por fin $e_{trans-change}(j)$ es la cantidad de energía consumida en el estado j, la cual puede ser expresada como:

$$E_{trans-transition} = \sum_{j=1}^{n} N_{trans-change}\, e_{trans-change}(j)$$

$$e_{trans-change}(j) = T_{init-end}(j)\left(\frac{P_{init}(j) + P_{end}(j)}{2}\right) \quad (4)$$

$$e_{trans-change}(j) = V_{tr} T_{init-end}(j)\left(\frac{I_{init}(j) + I_{end}(j)}{2}\right) \quad (5)$$

6.3 Modelo de energía del sensor

El módulo sensor está compuesto, de una parte, por dispositivos específicos que captan datos de variables como temperatura, humedad, niveles de radiación, amoniaco, metano, entre otros y de otra, por conversores análogos y digitales. El consumo de energía para este módulo resulta de la ejecución de operaciones como el muestreo de la señal, la conversión análoga/digital o la modulación de la señal. El módulo sensor puede operar en modo aleatorio o en modo periódico, de acuerdo con la configuración predeterminada por el administrador del sistema. Cabe señalar que, en general, se prefiere la operación de este módulo en modo periódico. Suponiendo que los consumos de energía en las operaciones abrir (*open*), cerrar (*close*) son constantes, la energía consumida por el sensor puede ser expresada como:

$$E_{sensor} = E_{on-off} + E_{off-on} + E_{sensor-run}$$

$$E_{sensor} = N(e_{on-off} + e_{off-on} + V_s I_s T_s) \quad (6)$$

donde e_{on-off} es el consumo de energía por unidad de tiempo del sensor durante la operación de apagado; e_{off-on} es el consumo de energía por unidad de tiempo del sensor durante la operación de encendido; $E_{sensor-run}$ es el consumo de energía durante la operación de sensado; V_s e I_s corresponden al voltaje y corriente de trabajo normales del sensor; T_s es el intervalo de tiempo de la toma de datos del sensor y N es el número de veces que el sensor se cierra y se abre para tomar medidas.

6.4 Modelo completo de energía para el nodo

Es importante señalar que en los nodos tanto el procesador como el radio y los componentes sensores deben trabajar de forma cooperativa para llevar a cabo una tarea computacional; este hecho implica la existencia de una relación mutua entre todos los componentes y, por ende, esta relación afecta los consumos de energía del nodo completo. Por consiguiente, en los cálculos de evaluación del desempeño no debe considerarse el sistema linealmente independiente, sino que se deben tener en cuenta cada uno de los componentes para el cálculo completo de la energía consumida por el nodo.

7. ARQUITECTURA PROPUESTA

La figura 1 muestra la primera arquitectura propuesta para lograr reprogramar la RIS usando agentes inteligentes (Ovalle, Restrepo y Montoya, 2010). Lo primero que se realizó fue un reemplazo de la estación base clásica de las RIS por un sistema embebido especial (Sheevaplug, 2011), que es tal vez el computador más pequeño y barato del mercado (alrededor de 100 dólares). En este sistema embebido se ejecuta TinyOS, OpenJDK como versión de Java y Equinox (McAffer, VanderLei y Archer, 2010). Adicionalmente se tiene instalado, como plataforma de agentes inteligentes, el AFME (Agent Factory Micro Edition). En el sistema embebido se despliegan los agentes inteligentes como módulos llamados *bundles*. Además, en los nodos se tiene una pequeña máquina virtual que permite tener un esquema de tableros o tuplas, similar a la propuesta de Agilla (Fok, Roman y Lu, 2009b).

Los nodos sensores han sido desplegados con una programación inicial y se tiene control sobre las variables por medir y los parámetros específicos, como los tiempos de muestreo, la calibración, entre otros. En nuestra propuesta los agentes móviles migran desde el sistema embebido y pueden entrar en contacto con un nodo definido, de acuerdo con su identificación y, por ejemplo, modificar un arreglo de código que posee el nodo. La forma de ejecutar este proceso es similar a la usada en MASPOT (ver sección 5), en cuanto al almacenamiento de los arreglos de código, su interacción y actualización. Sin embargo, en nuestro enfoque son los agentes inteligentes móviles los que se encargan del proceso de modificación de los parámetros y de la reprogramación de los nodos. Los agentes móviles son simples, ellos van, modifican el valor y se destruyen luego de realizar su tarea. Esto con el único objetivo de tener el mayor gasto en consumo energético cuando se transmiten los datos desde la estación base hasta el nodo sensor. De esta forma, se pretende disminuir el consumo innecesario de energía en el proceso de transmisión del sensor inalámbrico a la estación base. Puede decirse que es una solución computacional eficiente donde se hace el mejor esfuerzo para evitar confirmaciones y evitar retransmisiones innecesarias desde los nodos.

La justificación en el uso de las herramientas base para el desarrollo de esta solución radica en que OSGi –Open Services Gateway Initiative– (OSGi Alliance, 2011) facilita el desarrollo de componentes modulares, extensibles, abiertos, flexibles, reusables

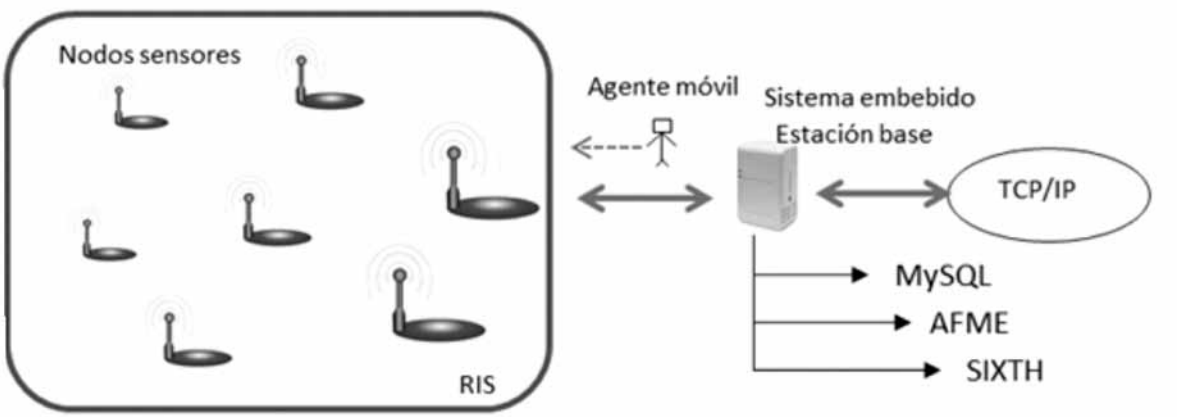

Figura 1. Arquitectura inicial de la solución propuesta, usando el sistema embebido y agentes móviles inteligentes para reprogramar la RIS

y escalables; SIXTH (SIXTH, 2011) promueve la modularidad, escalabilidad, reúso y heterogeneidad en las redes, siendo un marco de desarrollo de software para RIS de código libre basado en Java. Finalmente, el entorno de desarrollo de agentes AFME (Agent Factory Micro Edition) fue escogido debido a que facilita la creación de agentes móviles inteligentes en dispositivos con recursos limitados como son las RIS.

A continuación se presenta una breve descripción de estas tres herramientas.

7.1 Plataforma OSGi

La plataforma OSGi (Open Services Gateway Initiative) se caracteriza por ser un sistema modular y una plataforma de servicios basados en Java que implementa un modelo de componentes dinámicos y modulares. Las aplicaciones o componentes desarrollados pueden ser instalados de forma remota, iniciados, detenidos o actualizados sin requerir el reinicio del sistema. Por su parte, la administración de las clases de Java y los paquetes son especificados en gran detalle y se trabaja con el concepto de módulos. OSGi hace a Java modular y permite desarrollos de software completamente distintos a los tradicionales con Java (McAffer, VanderLei y Archer, 2010).

7.2 SIXTH

SIXTH es un entorno de desarrollo de software, basado en Java que promueve la modularidad, escalabilidad, reúso y heterogeneidad en las RIS y su interconexión con la web (SIXTH, 2011). El principal objetivo de SIXTH es ofrecer posibilidades de un desarrollo rápido de aplicaciones con diferentes tecnologías en RIS. El entorno SIXTH está construido usando el entorno de desarrollo OSGi, lo cual facilita la creación de componentes reutilizables, flexibles, abiertos y modulares. La arquitectura de SIXTH está formada por tres capas: adaptadores, API y la capa de servicio. SIXTH soporta el reúso de componentes y módulos; además ofrece la posibilidad de múltiples abstracciones e inteligencia embebida usando la plataforma de desarrollo de agentes AFME (Muldoon *et al.*, 2006).

7.3 Plataformas AF (Agent Factory) y AFME (AF Micro Edition)

La plataforma Agent Factory (AF) fue desarrollada en CLARITY (Centre for Sensor Web Technologies) de la University College of Dublin, Irlanda (Collier, 2001). AF está compuesta por una colección de herramientas, plataformas y lenguajes que soportan el desarrollo y despliegue de sistemas multiagente. El entorno de desarrollo está dividido en dos plataformas, la AF que soporta el desarrollo de agentes inteligentes en computadores de escritorio, estaciones de trabajo y servidores y la AFME (Agent Factory Micro Edition) que está diseñada para dispositivos embebidos, como teléfonos móviles, PDA y sensores inalámbricos (Muldoon, 2008). Ambas cumplen con los estándares de FIPA (Foundation for Physical Inteligent Agents) (FIPA, 2011) y usan AFAPL (Agent Factory Agent Programming Language) como intérprete.

8. ANÁLISIS EXPERIMENTAL

El amoniaco es un gas altamente tóxico, usado de forma permanente en la industria alimentaria donde es crítica e indispensable la conservación de los alimentos mediante el control de la temperatura. Es frecuente el uso de redes de amoniaco desplegadas sobre toda el área empresarial buscando producir temperaturas aptas para la manipulación de los productos. En caso de una fuga de amoniaco pueden ocurrir varios eventos: 1) Que se contaminen los alimentos, en cuyo caso no se tienen investigaciones sobre el efecto del consumo de amoniaco por parte de los seres humanos y su efecto en la salud a largo plazo. De acuerdo con lo anterior, es de vital importancia producir alimentos que tengan trazabilidad y se garantice que durante el proceso de producción, enfriamiento y almacenamiento no se hayan contaminado, de forma tal que se tengan alimentos libres de amoniaco para el consumo humano; 2) Que

existan fugas en la red de amoniaco de 30 a 50 ppm (partes por millón), en este caso se debe ordenar una evacuación permanente de los empleados que laboran allí, hasta controlarlas, porque el amoniaco causa daños irreversibles en las vías respiratorias y, en el peor de los casos, produce la muerte. Cabe señalar también que el amoniaco es un material explosivo.

Para validar la arquitectura propuesta, desplegamos una RIS que permite medir niveles de amoniaco (NH_3). La red está compuesta por 40 nodos sensores inalámbricos, distribuidos a lo largo de una planta de producción de alimentos cárnicos. Es importante señalar que el número de nodos de la RIS no está limitado por la aplicación específica ni por la plataforma utilizada. En el entorno experimental y en el diseño de pruebas se usaron para este caso de estudio nodos para medir gases (p. ej. amoniaco), sin embargo, se puede generalizar esta arquitectura y emplear otro tipo de sensores, como los de luminosidad o de temperatura.

El primer sistema de enfriamiento se encuentra a 2° C, allí se almacena toda la materia prima que se recibe (o sea, cortes de carne que deben permanecer en sistemas de enfriamiento permanente durante todo el proceso de producción). Para esta primera cámara frigorífica se tienen 6 sensores de amoniaco que permiten medir fugas tempranas. Luego, se observa el área de producción (o salas blancas) donde se producen y conservan las carnes frías. En esta segunda sala se situaron otros tres sensores de amoniaco. De forma similar, a lo largo del proceso de producción, se tiene un total de 40 sensores dedicados a medir niveles de amoníaco y que funcionan de forma permanente en toda la planta de producción.

Los sensores detectan concentraciones de amoniaco de 10 ppm a 100 ppm, con un error de 5 ppm. Cuando la fuga de amoniaco en un punto definido es de 30 ppm, el sensor envía un mensaje de alarma que es visualizado en una baliza ubicada en el centro de control de la empresa y se activa un botón amarillo en la ventana de control del software, mostrando el sitio exacto y estado de la fuga. En este caso, el área de operaciones puede tomar medidas tales como revisar el sistema o parar en un momento definido el flujo de amoniaco. Si el sensor detecta 50 ppm, un conjunto de mensajes de emergencia se activa y se ordena la evacuación del lugar donde se ha detectado la fuga. La figura 2 muestra la distribución de algunos de los sensores de amoniaco en la empresa (ver círculos verdes), sobre los cuales se lleva a cabo el experimento de reprogramación de parámetros y reconfiguración de nodos.

En la versión inicial de la arquitectura, se tenía solo una estación base, conectada a un servidor Linux instalado en el sistema embebido. En el sistema embebido se instaló la base de datos MySQL 5.0, el servidor web Apache 2 y el servidor de aplicaciones Tomcat, al igual que Equinox como servidor de OSGi. Con esta arquitectura se realizaron pruebas similares a las descritas dentro del proyecto MASPOT; la diferencia radica en que en nuestro sistema se tiene instalado el entorno de desarrollo AFME para la creación de agentes móviles y un conjunto de *bundles*, creados con la herramienta OSGi, que permiten ejecutar una serie de tareas sobre los nodos escogidos en la RIS. La estructura de la arquitectura inicial, mostrada en la figura 1, permitió hacer las primeras pruebas del uso de agentes móviles inteligentes dentro de RIS, con el fin de obtener medidas claras de los consumos energéticos. Lo anterior permitió analizar la viabilidad y el rendimiento en la creación de nodos sensores autónomos e inteligentes (Piedrahíta, Montoya y Ovalle, 2010).

Es importante señalar que, teniendo en cuenta el tamaño de la red, fue necesario segmentarla, utilizando diferentes estaciones base, por ejemplo, la red de sensores de amoniaco debía estar interconectada al sistema de alarmas de la empresa. Además, el hecho de tener una sola estación base se volvía un cuello de botella, porque todos los mensajes enviados por los nodos debían llegar a un solo punto, creando congestión en la red debido al alto número

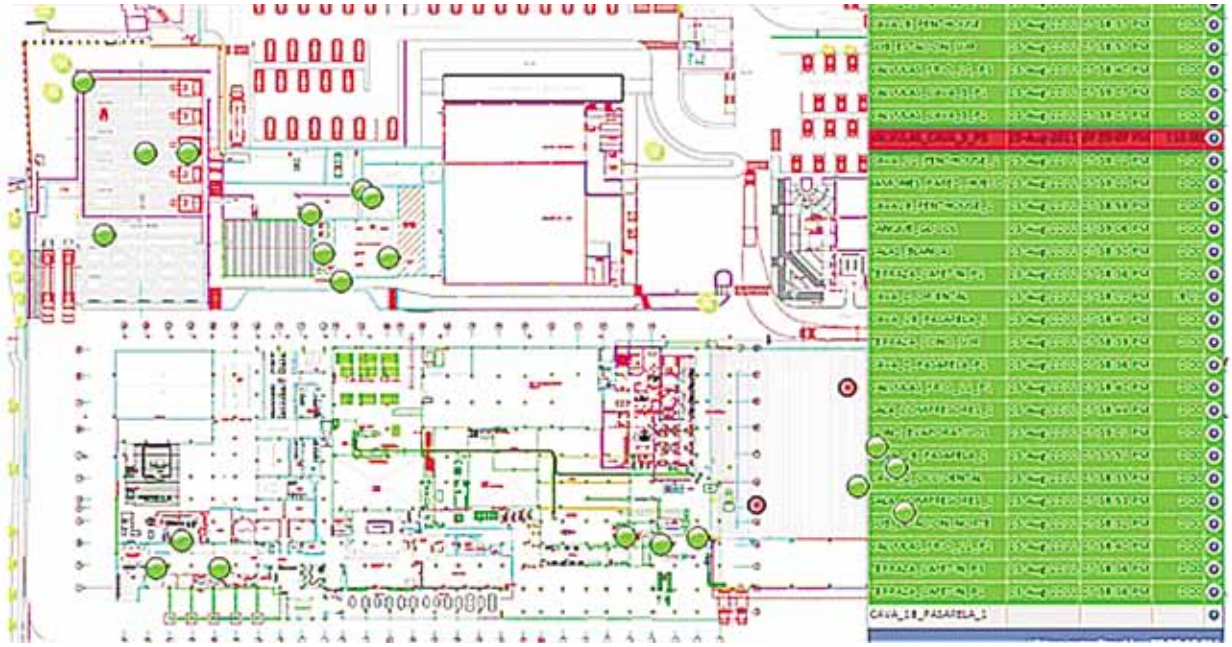

Figura 2. Vista de la planta de producción de alimentos cárnicos y la aplicación donde se aprecia la ubicación de algunos sensores (círculo verde)

de sensores (40 sensores de amoniaco, adicionales a 20 sensores de presión para activar las alarmas de evacuación de la empresa). Se decidió entonces reemplazar la estación base clásica por un sistema embebido. Las estaciones base ejecutan las tareas fundamentales para un conjunto de nodos definido, en este caso, el criterio de selección era la cercanía entre sensores, dado el tamaño de la empresa. Dentro de las nuevas estaciones base, se instaló el software para agentes inteligentes soportado por AFME y OSGi y se iniciaron las primeras pruebas base para reprogramar, reconfigurar y recalibrar algunos de los nodos sensores dentro de la red. La figura 3 muestra en detalle la arquitectura mejorada propuesta para el conjunto de nodos sensores empleados y su interconexión con las estaciones base, conformando así una RIS heterogénea, es decir, compuesta por distintos tipos de nodos sensores.

9. EVALUACIÓN DEL DESEMPEÑO DEL SISTEMA

En el sistema propuesto, la comunicación entre agentes se hace por medio de la estructura de datos "tuple-spaces" (Gelernter, 1985), la cual es similar al modelo propuesto por Agilla (Fok, Roman y Lu, 2009b). Un arreglo de código es un conjunto ordenado de valores pertenecientes a alguno de los siguientes tipos: booleano, string, array, byte, double, int y long. El espacio de arreglos de código se encuentra dentro de cada nodo y no necesariamente es igual para todos los nodos. Los arreglos de código se pueden acceder de forma remota por medio de la estación base, que envía y recibe respuestas. Para este caso de estudio, solo se envían comandos de reconfiguración, recalibración o reprogramación de algún parámetro de interés al

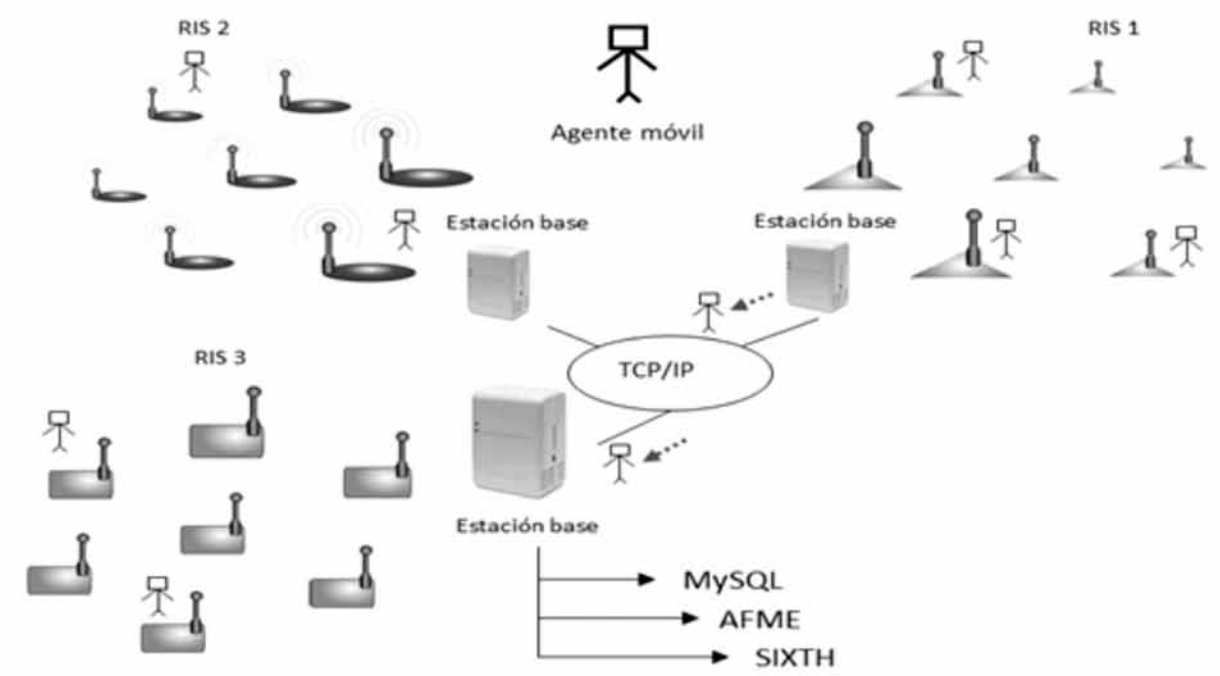

Figura 3. Arquitectura mejorada de interconexión de diferentes RIS heterogéneas usando el sistema embebido como estación base para cada una de las redes

nodo específico. En cuanto a las operaciones sobre un arreglo de código, pueden ser de lectura de los valores, escritura de nuevos valores o ejecución de un comando definido desde la estación base. Estas tres operaciones pueden ser ejecutadas en forma remota, de forma similar a como lo hace el sistema Agilla. En lo concerniente a los agentes móviles programados dentro de nuestro sistema con AFME, cabe resaltar que pueden usar estas operaciones para interactuar con otros nodos dentro de la red. Sin embargo, en los experimentos hechos no se tiene interacción entre nodos, sino interacción entre el espacio de arreglos de código de la estación base y los comandos que se envían, con el espacio de arreglos de código de los nodos.

9.1 Evaluación del costo de migración

El primer parámetro para reconfigurar dentro de los nodos de prueba fue el tiempo de muestreo de los sensores de amoniaco. Originalmente el tiempo de muestreo de cada sensor era de 1 minuto (utilizando reprogramación y esquema de arreglos de código). Sin embargo, mediante agentes móviles inteligentes que salen de la estación base o sistema embebido, se busca modificar el tiempo de muestreo de amoniaco a 30 segundos. En este escenario, se midió la corriente promedio consumida por cada sensor, no solo durante la llegada del agente móvil sino también la asociada al proceso de reconfiguración, recurriendo al modelo matemático descrito en la sección 6. Así mismo, se midió el tiempo de convergencia de la red, que corresponde al tiempo necesario para que el nodo se reconfigure de forma autónoma con respecto al nuevo tiempo de muestreo.

Tabla 1. Consumo promedio de corriente reconfigurando el parámetro tiempo de muestreo en 10 sensores de amoniaco

Nodo	Corriente eléctrica consumida (mA)
1	0,4891
2	0,4912
3	0,3812
4	0,3914
5	0,3216
6	0,4410
7	0,4120
8	0,4317
9	0,3945
10	0,4513
Promedio total	0,4205

9.2 Evaluación del costo energético

Los costos energéticos en reprogramación fueron bajos (se utilizaron baterías en los nodos de 720 mA). Si observamos la tabla 1, se puede concluir que la corriente consumida por la batería en el proceso de reprogramación del cambio de periodo de muestreo en cada nodo fue muy bajo, en promedio de 0,4205 mA, lo que implica un gasto de energía de solo el 0,05 % de la energía disponible en el nodo sensor. Este resultado demuestra que el uso de agentes inteligentes móviles para ejecutar la tarea de reprogramación o reconfiguración es viable desde el punto de vista de consumo energético en los nodos de la RIS, comparado con los métodos de reprogramación convencionales que no usan técnicas de inteligencia artificial.

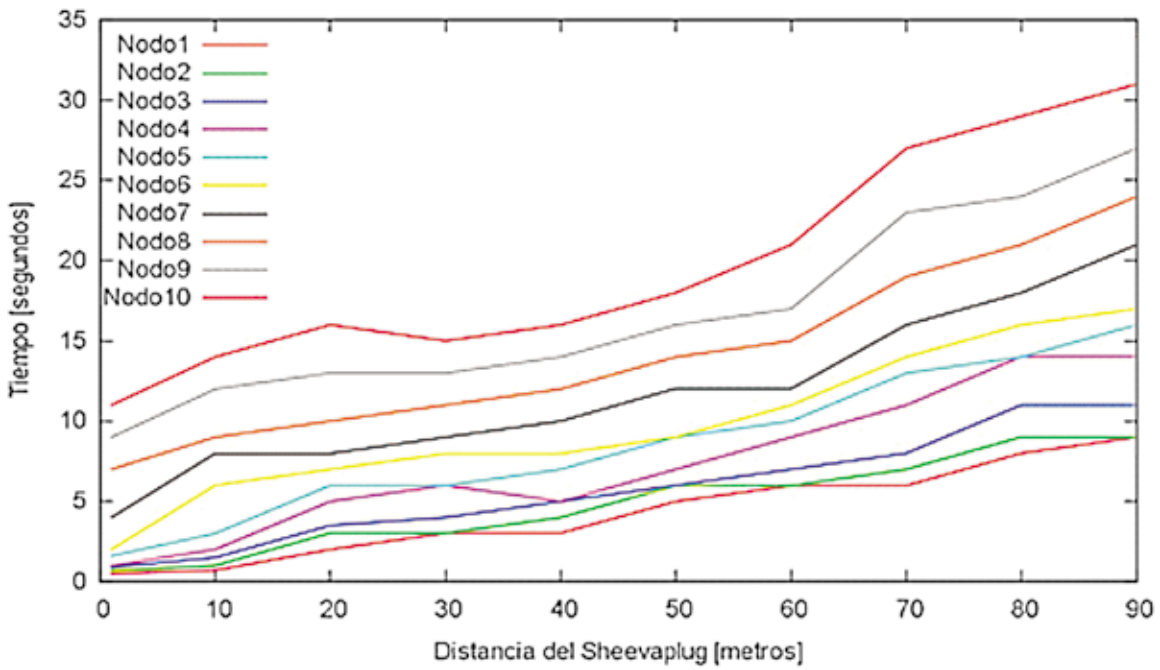

Figura 4. Tiempo de convergencia en la reconfiguración autónoma de un parámetro contra distancia

9.3 Evaluación del tiempo de convergencia

El segundo experimento consistió en medir el tiempo de convergencia en la reconfiguración de un parámetro, respecto a la distancia a la cual se encuentra el sensor inalámbrico de su estación base. Para este caso, se tomaron los sensores y se pusieron a diferentes distancias, de modo tal que hubiese obstáculos y que estuviesen alejados de la estación base, tal cual es el caso real cuando se instala una RIS. En la figura 4 se puede observar que el tiempo de convergencia máximo fue de unos 35 segundos, es decir, el nodo más alejado de la estación base logró en este tiempo el cambio de reconfiguración del parámetro (o sea, el tiempo de muestreo de la variable amoniaco). Este resultado muestra que los tiempos de convergencia para que la RIS haga el cambio de un parámetro y su reconfiguración de forma autónoma son muy bajos comparados con los de las redes convencionales y, en consecuencia, la arquitectura de RIS propuesta puede usarse para lograr la reconfiguración de redes más complejas.

10. CONCLUSIONES Y TRABAJO FUTURO

El empleo de agentes móviles inteligentes para reprogramar y reconfigurar RIS de forma autónoma es viable y eficiente, dada la disminución en los costos de energía y los tiempos de convergencia que se logran al comparar los resultados con los de otros enfoques, tales como el utilizado en el proyecto MASPOT y las RIS convencionales. En efecto, usando una arquitectura basada en agentes móviles inteligentes integrados con RIS, se determinó que el punto clave consiste en reducir el consumo de energía producto de las confirmaciones y retransmisiones innecesarias de datos y procedimientos desde los nodos sensores hasta la estación base, porque se transmiten datos que son previamente analizados por los agentes inteligentes. Lo anterior representa disminución de costos energéticos y ahorros en los tiempos de convergencia de los nodos sensores de más o menos el 45 % en comparación con los consumos de energía de las RIS convencionales.

La arquitectura propuesta muestra cómo reemplazando las estaciones base convencionales de las RIS por sistemas embebidos se permite mejorar el rendimiento de la red, en cuanto a los consumos de energía y ahorros de tiempo en la reprogramación y reconfiguración de los nodos sensores, de forma autónoma. Esto ocurre porque la nueva estación base tiene acceso a recursos de energía, procesamiento y software más nutridos que la estación base convencional. Al realizar la comparación con propuestas como MASPOT y MAWSN se observa que MASPOT usa computadores de escritorio, lo cual hace inviable desplegar este tipo de red en producción, y MAWSN basa sus resultados en simulaciones, no en experimentos reales como los descritos. Para la experimentación realizada se utilizaron las herramientas AFME, para creación de agentes móviles inteligentes; así mismo, OSGi y SIXTH que proveen modularidad, reúso de componentes, escalabilidad, entre otros, en RIS heterogéneas. La reconfiguración y reprogramación, de forma autónoma, fue validada en una RIS compuesta por sensores para detección de gas amoniaco en una empresa de cárnicos, la cual se encuentra en funcionamiento. Lo anterior busca transferir los resultados de la investigación a casos reales de RIS desplegadas y en funcionamiento.

Cabe señalar que en la revisión de la literatura no se hallaron referencias a estudios hechos con RIS para medir fugas de amoníaco en entornos industriales a mediana escala, como es el caso de estudio de este trabajo. Tampoco se reportan referencias de RIS que usan sistemas embebidos como estaciones base que ejecutan agentes inteligentes. Esto hace que esta investigación sea pionera en el área y se diferencia de modo notable de otros enfoques propuestos por la comunidad científica.

Como trabajo futuro, se pretende realizar nuevos experimentos para evaluar el desempeño de agentes móviles en donde las operaciones de reprogramación, reconfiguración y ejecución de

tareas se puedan invocar de nodo a nodo en lugar de nodo a estación base, con el fin de medir los costos de migración, consumo energético y tiempos de convergencia. Además, se pretende recurrir a otras métricas, tales como cobertura, escalabilidad, despliegue, tiempo de respuesta, tiempo de vida del sensor, efectividad del muestreo, seguridad, confiabilidad, conectividad, calidad del servicio, con el fin de evaluar en forma global el desempeño de la integración de agentes inteligentes y RIS. Finalmente, cabe resaltar que nuestro principal desafío es el desarrollo de un modelo completo de sistema híbrido inteligente que permita acercarnos a una RIS autónoma, la cual debe comportarse como un sistema inteligente en su totalidad.

AGRADECIMIENTOS

Los autores agradecen a los profesores Gregory O'Hare y Michael O'Grady por las valiosas sugerencias realizadas al trabajo de investigación durante la estancia doctoral en CLARITY (Centre for Sensor Web Technologies) de University College at Dublin, Ireland, en el 2010. Igualmente a Colciencias por el soporte financiero brindado al proyecto de investigación titulado "Desarrollo de un modelo de sistema híbrido inteligente para detección y control remoto de variables físicas usando redes de sensores inalámbricas distribuidas" en la vigencia 2010-2012.

REFERENCIAS

Akyildiz, I. F.; Melodia, T.; and Chowdhury, K. R. (2007). "A survey on wireless multimedia sensor networks". *Computer Networks*, vol. 51, No. 4, pp. 921-960.

Akyildiz, I. F.; Pompili, D. and Melodia, T. (2004). "Challenges for efficient communication in underwater acoustic sensor networks". *SIGBED Review,* vol. 1, No. 2, pp. 3-8.

Akyildiz, I. F.; Su, W.; Sankarasubramaniam, Y. and Cayirci, E. (2002). "Wireless sensor networks: A survey". *Computer Networks,* vol. 38, No. 4 (March), pp. 393-422.

Boulis, A. and Srivastava, M. B. (2002). *A framework for efficient and programmable sensor networks*. Proceedings of 2002 IEEE Open Architectures and Network Programming. (OpenArch 2002). New York, NY (28-29 June), pp. 117-128.

Chen, M.; Kwon, T; Yuan, Y. and Leung, V. C. M. (2006). "Mobile agent based wireless sensor networks". *Journal of Computers,* vol. 1, No. 1 (April), pp. 14-21.

Cheong, E. (2007). *Actor-oriented programming for wireless sensor networks*. EECS Department, University of California, Berkeley. Technical report No. UCB/EECS - 2007 -112 (30 August). 138 p.

Collier, R. W. *Agent factory: A framework for the engineering of agent-oriented applications*, Ph.D. Thesis, University College Dublin, Ireland, 2001.

Crossbow Technology, Inc. *Mote in-network programming user reference*. TinyOS document, [online] 2003 [consulted on February 25, 2011] Available in: <http://www.tynos.net/tinyos-1.x/doc/Xnp.pdf>

Culler, D.; Estrin, D. and Srivastava, M. (2004). "Guest editors' introduction: Overview of sensor networks". *Computer,* vol. 37, No. 8, pp. 41-49.

FIPA –The Foundation for Intelligent Physical Agents– (2011) [online]. IEEE Computer Society. Available in: <http://www.fipa.org/>.

Fok C-L.; Roman G-C. and Lu, C. *A mobile agent middleware for wireless sensor networks*. 2009a. [consulted on September 30, 2009] Available in: <http://mobilab.wustl.edu/projects/agilla/>.

Fok, C-L.; Roman, G-C. and Lu, C. (2009b). "*Agilla: A mobile agent middleware for self-adaptive wireless sensor networks". ACM Transactions on Autonomous and Adaptive Systems,* vol. 4, No. 3 (July), pp.1-26.

Gelernter, D. (1985). *Generative communication in Linda*. ACM Transactions on Programming Languages and Systems, vol. 7, No. 1 (January), pp. 80-112.

Hui, J. and Culler, D. (2004). *The dynamic behavior of a data dissemination protocol for network programming at scale*. ACM SenSys'04, Baltimore, MD (3-5 November), pp. 81-94.

Levis, P. and Culler, D. (2002). *Maté: A tiny virtual machine for sensor networks*. Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS' 02), San Jose, CA (6-10 October), pp. 85-95.

Li, M. and Liu, Y. (2007). *Underground structure monitoring with wireless sensor networks*. Proceedings of the 6th International Conference on Information Processing in Sensor Networks, Cambridge, MA (25-27 April), pp. 69-78.

Liu T. and Martonosi M. (2003). *Impala: A middleware system for managing autonomic, parallel sensor systems*. Proceedings of the Ninth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP' 03), pp. 107-118.

Lopes, R.; Assis, F. and Montez, C (2011). *MASPOT: A mobile agent system for Sun SPOT*. Proceedings of the 2011 Tenth International Symposium on Autonomous Decentralized Systems (ISADS), Tokyo and Hiroshima (23-27 March), pp. 25-31.

Lu, F.; Xu, G. Z. and Ying, R. D. (2005). "Power consumption simulation model based on the working status of Intel PXA250 processor. *Control & Automation,* vol. 21, No. 1, pp. 131-132.

McAffer, Jeff; VanderLei, Paul and Archer, Simon. *OSGi and Equinox: Creating highly modular Java systems*. Addison-Wesley Professional, 2010. 460 p.

MEMSIC [online] [consulted on February 25, 2011] Available in: <http://www.memsic.com/products/wireless-sensor-networks/wireless-modules.html>.

Muldoon, C. (2008). *An agent framework for ubiquitous services*. Ph.D. Thesis, School of Computer Science and Informatics, University College Dublin, Ireland.

Muldoon, C.; O'Hare, G. M. P.; Collier, R. W. and O'Grady, M. J. O. (2006) *Agent factory micro edition: A framework for ambient applications*. In: Intelligent agents in computing systems, volume 3993 of Lecture Notes in Computer Science. Reading, UK: Springer 2006 (28-31 May), pp. 727-734.

OSGi Alliance [online] 2011. [consulted on February 25, 2011] Available in: <http://www.osgi.org/About/HomePage>.

Ovalle, Demetrio; Montoya, Alcides y Muñoz, Tatiana. *Análisis de métricas para la evaluación del desempeño de WSNs enriquecidas con agentes inteligentes*. En: Tendencias en ingeniería de software e inteligencia artificial, vol. 4. Medellín, Universidad Nacional de Colombia, 2001, pp. 139-146.

Ovalle, Demetrio; Restrepo, Diana and Montoya, Alcides; (2010) *Chap. 4: Artificial intelligence for wireless sensor networks enhancement*. In: Smart wireless sensor networks. Hoang Duc Chinh & Yen Kheng Tan (eds.), pp. 73-81, 2010.

Piedrahíta, Alejandro; Montoya, Alcides and Ovalle, Demetrio. *Performance evaluation of an intelligent agents based model within irregular WSN topologies*. In: Innovations in computing sciences and software engineering. Springer Science, 2010, pp. 571-576.

Römer, K. and Mattern, F. (2004). "The design space of wireless sensor networks". IEEE *Wireless Communications,* vol. 11, No. 6 (December), pp. 54-61.

Russell, S. J., and Norving, P. *Artificial intelligence: A modern approach*. 2nd ed. Englewood Cliffs, NJ: Prentice Hall, 2003.

Sheevaplug [online] [consulted on February 25, 2011] Available in: <http://www.plugcomputer.org]>

Simon, D.; Cifuentes, D.; Cleal, D.; Daniels, J. and White, D. (2005). *The squawk Java virtual machine: Java on the bare metal*. Proceedings of the 20th Object-Oriented Programming Systems, Languages and Applications (OOPSLA 2005). San Diego, CA (16-20 October).

SIXTH. [online] [consulted on: February 28, 2011] Available in: <http://www.clarity-centre.org/SIXTH/index.php>

Sycara, K. (1998). "Multiagent systems. *AI Magazine,* vol. 19, No. 2, pp. 79-92.

TinyOS [Online]. [consulted on: February 25, 2011] Available in: <http://www.tynos.net>

Tompkins, R.; Jones, T. B.; Nertney, R. E.; Smith, C.E. and Gilfeather-Crowley, P. (2011), *Reconfiguration and management in wireless sensor networks*. 2011 IEEE Sensors Applications Symposium (SAS). San Antonio, TX (22-24 February), pp. 39-44.

Toumpis, S. and Tassiulas, L. (2006). *Optimal deployment of large wireless sensor networks*. IEEE Transactions on Information Theory, vol. 52, No. 7, pp. 2935-2953.

Tynan, R.; O'Hare G. M. P.; Marsh D. and O'Kane D. (2005). *Multi-agent system architectures for wireless sensor networks*. Proceedings of the International Conference on Computational Science (ICCS 2005), Atlanta, GA (22-25 May), vol. 3, pp. 687-694.

Wang, X.; Xiang. and Hu, B. (2009). "Evaluation and improvement of an energy model for wireless sensor networks. *Chinese Journal of Sensors and Actuators,* vol. 22, No. 9, pp. 1319-1321,

Wooldridge, M. *An introduction to multi-agent systems*. 2nd ed. Chichester, UK: John Wiley & Sons, 2009.

Yick, J.; Mukherjee, B. and Ghosal, D. (2008). Wireless sensor network survey. *Computer Networks,* vol. 52, No. 12, pp. 2292-2330.

Zhou, D. and Gao, J. (2010). *Maintaining approximate minimum Steiner tree and k-center for mobile agents in a sensor network*. Proceedings of the IEEE INFOCOM 2010, San Diego, CA (March 15-19), pp. 1-5.