# Nirvana: Una herramienta para la monitorización de procesos que emplea redes inalámbricas de sensores y características del protocolo SNMP

L. De Llano y R. González

Universidad Simón Bolívar, Caracas-Venezuela

**RESUMEN**

*Este trabajo describe a Nirvana una herramienta que permite la recolección, almacenamiento y visualización de la información de datos sensados o provenientes de la monitorización de sistemas físicos, mediante una red inalámbrica de sensores (RIS) que usa el protocolo SNMP. Nirvana también aprovecha las características del protocolo SNMP y de la recolección de datos de los sistemas de gestión de redes, que son muy similares a los perfiles de uso de la información que recolectan las RIS, para integrar a Cacti, una herramienta empleada para el almacenamiento y visualización de datos de gestión de redes, que en nuestro caso es usada como front end de la red, ya que es una herramienta RRDTools bastante completa que permite implementar fácilmente los servicios de visualización de los datos que han sido recolectados por la RIS .*

**ABSTRACT**

*This work describes Nirvana, a tool that allows the collection, storage and visualization of sensing or monitoring data from physical systems, using a wireless sensor network that implement the SNMP protocol. Nirvana takes advantage of some of the characteristics of the SNMP protocol and the data collection facilities from network management tools, which have a very similar profiles of use, compared with the wireless sensor network data gathering features, to integrate Cacti, a network management and visualization tool, as the Nirvana front end, because former one is an interesting RRDTools that can be used for storing, querying and visualizing of any data that was being collected by the RIS.*

**Keywords**: Redes Inalámbricas de Sensores, SNMP, Redes Inalámbricas.

## 1. Introducción

Una red inalámbrica de sensores (RIS) es una red compuesta por una cantidad de pequeños nodos sensores o *motes* que son colocados para observar un fenómeno de interés [ASSC02]. En este tipo de red, cada nodo sensor es un pequeño computador con funcionalidades básicas, compuesto de una unidad de procesamiento o microcontrolador, memoria, sensores, un dispositivo de comunicación inalámbrica y una fuente de poder [MSK*04]. En una red de sensores usualmente existe un nodo especial llamado *sink*, que posee menores limitaciones de consumo de potencia, así como dispositivos de comunicación más complejos y potentes, lo que le permite cumplir el rol de *gateway* entre los nodos sensores y los otros componentes de la arquitectura de la red.

Utilizar redes para monitorizar procesos permite reducir costos de operación, ya que una vez que los mismos están siendo monitorizados, la detección de fallas puede ocurrir tan pronto como éstas ocurren. Adicionalmente Miorandi, Uhlemann, Vitturi y Willig [MUVW07] comentan que las redes inalámbricas proveen un enfoque natural hacia la comunicación con equipos móviles, donde los cables al moverse están en constante riesgo de romperse. Las redes que se utilizan para monitorizar procesos, y en especial procesos industriales, deben tener muchas conexiones para hacer seguimiento a cada una de las etapas de los procesos involucrados, si estas redes llegan a ser inalámbricas, se puede llegar a obtener una reducción significativa de los costes de instalación y mantenimiento de la red.

En la monitorización de procesos puede ser de mucha ayuda el uso de redes inalámbricas de sensores ya que a partir de los valores recolectados por éstas y según los objetivos de desempeño que se quieran cumplir, es posible realizar tareas de control al modificar el comportamiento del sistema, además se podrá garantizar la calidad, seguridad y eficiencia de los procesos, en cada una de sus fases. Es importante que los datos sean recolectados y analizados con la frecuencia suficiente para poder detectar a tiempo cualquier falla del proceso o incluso para llagar a estar en capacidad de predecirlas y manejarlas antes de que ocurran o de que se vuelvan críticas, esto puede lograrse al estudiar las estadísticas y tendencias de los datos recolectados.

## 2. Antecedentes

SNMP es un protocolo de gestión de red que ya ha sido integrado en redes inalámbricas de sensores anteriormente, tal como se describe en los trabajos de Hongseok, Kugsang y Deokjai [HKD06] y Berruti, Denegri y Zappatore [BDZ07] específicamente para la recolección de datos de sensores hasta el *sink*. Otros trabajos han sido elaborados en los que se recolectan los datos de los sensores mediante algoritmos propios de las RIS y luego de recolectados los datos se muestran a través de interfaces que emplean SNMP, tal como ocurre en el trabajo de Lim, Messina, Kargl, Ganguli, Fischer, y Tsang [LMK*08], en el que se usa un SNMP Proxy Agent, así como en el trabajo de Landaeta [Lan08], quien crea un mediador que transforma los datos de la RIS a formato SNMP para que estos se puedan visualizar en herramientas que aceptan mensajes en SNMP, como Cacti [Cac10].

El presente trabajo muestra un esquema de extremo a extremo en el que se usa una adaptación del protocolo SNMP desde el envío de datos que realizan los nodos sensores hasta su llegada al *sink*, lo que permite que la información salgan de la RIS en SNMP y puedan llegar a otras herramientas, en redes TCP/IP, como lo es Cacti, que recibe la información en formato SNMP para recolectar los datos y permitir que los usuarios de la, red a través de interfaces Web, puedan consultar y visualizar los datos colectados por la RIS de forma rápida y eficiente.

## 3. Descripción General de Nirvana

Como se mencionó anteriormente en la monitorización de procesos se quiere realizar medidas de ciertos valores físicos en un punto específico de un sistema, este es justamente el objetivo de Nirvana, que es una aplicación que permite realizar mediciones de parámetros de interés en el punto donde un nodo sensor ha sido ubicado. Nirvana es una aplicación que realiza una recolección de datos de forma continua, donde todos los valores recolectados son enviados al nodo *sink*, sin que se produzca ningún proceso de fusión de datos o cálculo de estadísticas en los nodos intermedios de la red.

Nirvana es un palabra asociada a creencias orientales que representa el estado superior del ser, liberado de preocupaciones y del sufrimiento, así como del ciclo de muerte y renacimiento. Con esta aplicación se pretende contar con una red inalámbrica confiable, que libere de preocupaciones a los administradores de los procesos que se estudian. Es por esta razón que se ha elegido el nombre Nirvana para este sistema.

Para las pruebas pilotos se ha seleccionado la recolección de dos parámetros: la temperatura interna del nodo sensor y el voltaje interno que puede ser empleado como un indicador de la cantidad de energía restante en el nodo y por tanto de su vida útil. Sin embargo, este programa se ha diseñado con la intención de ser fácilmente modificable para incluir cualquier otro dato que pueda ser de importancia en la monitorización de procesos, ya sea porque el nodo sensor está en la capacidad de realizar la medición de dicho valor a través de un sensor, o porque puede realizar una estimación a partir de otros datos obtenidos previamente.

Para utilizar la aplicación lo primero que se debe hacer es ubicar los nodos sensores en el lugar indicado, donde nos interesa realizar las mediciones. Para ubicar los nodos de la RIS se deben realizar los procesos de *Site Survey* y despliegue de la red, éstos puede ser desarrollados con ayuda de la herramienta Aunris [DG10]

Como se mencionó anteriormente uno de los objetivos que se busca alcanzar mediante la aplicación Nirvana es desarrollar una herramienta que pueda ser utilizada para realizar la monitorización de procesos a través de una red inalámbrica de sensores. Para poder comprender el funcionamiento de la herramienta desarrollada, primero se deben explicar algunos conceptos sobre la gestión de redes.

## 4. Gestión de Redes

En la aplicación Nirvana se quiere que el nodo *sink* sea capaz de pedir mediciones a los nodos sensores, y que los nodos sensores puedan reportar al sink el valor de las mediciones que realizan durante su operación. En los procesos de gestión de redes se definen operaciones para asistir en la toma y el intercambio de información de datos asociados a los nodos de la red. SNMP es un protocolo específico de gestión de redes que es utilizado en la aplicación Nirvana, para realizar el intercambio de información, de los datos capturados por los nodos sensores de la red, al *sink* donde pueden ser almacenados y visualizados. En la gestión de redes es muy importante poder conocer el estado de los nodos de la red, para identificar tanto fallas en concreto como situaciones especiales que con el tiempo puedan desencadenar comportamientos no deseados.

Debido a su tamaño y complejidad es difícil manejar las actuales redes de forma manual a causa de la cantidad y variedad de factores que pueden tener un impacto en el comportamiento de la red. Para atacar este problema se han desarrollado una serie de sistemas que permiten automatizar, al menos parcialmente, la recolección, el análisis e incluso el tratamiento de la información que permita identificar cuál es el estado de la red, de forma tal que se facilite la tarea de determinar qué acciones se pueden tomar para mejorarlo.

El propósito del manejo de redes es recoger información sobre el estado y comportamiento de los elementos de la red. La información a ser recogida puede incluir tanto información estática, relacionada a la configuración de los equipos, como información dinámica, relacionada a los eventos que ocurren en la red; llegando a cubrir también información estadística resumida de la información dinámica. Típicamente, cada elemento manejado en la red posee un módulo recolector o agente, responsable de registrar información local y transmitirla a una o más estaciones de manejo. Cada estación de manejo incluye software de aplicación de manejo de red, además de programas para comunicarse con los agentes. La información puede ser recolectada de forma activa, haciendo encuestas (*polling*) mediante una estación manejo, o de forma pasiva, mediante reportes de eventos provenientes de los agentes siguiendo el esquema de *traps*.

## 5. Recolección de los datos sensados mediante una RIS

Para este trabajo se propuso elaborar una herramienta para recolectar datos provenientes de una RIS. A partir de una investigación realizada sobre el protocolo SNMP se identificó que los esquemas de solicitud y consulta de in-

formación, mediante los comando *get, set* y *trap,* que utiliza el protocolo SNMP, se adaptan bastante viene a los esquemas de consultas y requerimientos de información de una RIS, es por ello que se decidió basar la implementación de la recolección de datos sensados por una RIS en este protocolo.

A continuación se describen las características generales del protocolo SNMP y las adaptaciones que se realizaron a este protocolo en el presente trabajo:

### 5.1 SNMP (Simple Network Management Protocol)

SNMP es un protocolo estándar de capa de aplicación para monitorizar redes IP. Fue diseñado para facilitar el intercambio de información de gestión entre dispositivos de red. SNMP define un conjunto de objetos que toda red debe cumplir y especifica que todas las operaciones en estos componentes son un efecto secundario de extraer o guardar información de estos objetos [CS91].

Una red manejada con SNMP consiste de tres componentes principales: dispositivos manejados, agentes y sistemas de manejo de red (NMSs por sus siglas en inglés, *Network-Management Systems*), los cuales se explican a continuación y se ilustran en la figura 1.

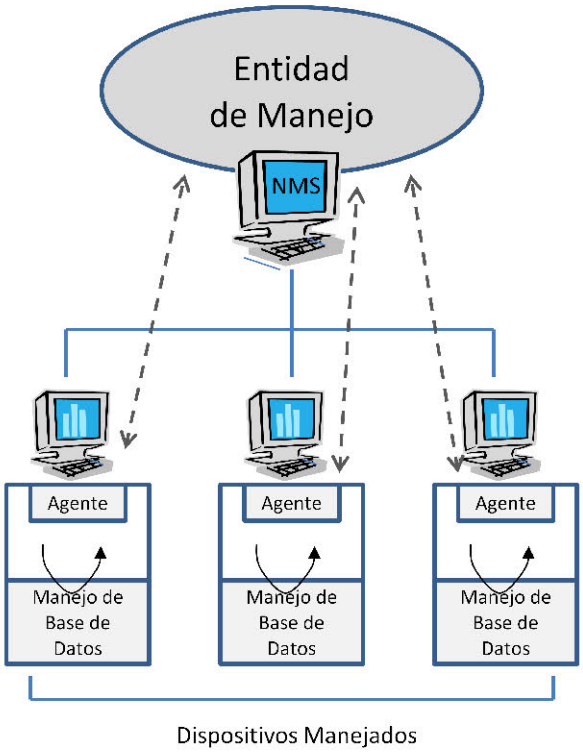


**Figura 1:** *Elementos de una red manejada con SNMP*

Un dispositivo manejado es un nodo de la red que contiene un agente SNMP, que reside en una red manejada. Los dispositivos manejados recolectan y guardan información de manejo de la red y hacen disponible esta información al NMSs usando SNMP. Los dispositivos manejados, llamados elementos de la red, pueden ser *routers, acces servers, switches, bridges, hubs, computer hosts*, impresoras o incluso nodos inalámbricos.

Un agente es un módulo de software de gestión de red que reside en un dispositivo manejado y que posee conocimiento local de información de manejo de red, y la traduce a una forma compatible con SNMP.

Un NMS ejecuta aplicaciones que monitorizan y controlan dispositivos de manejo de red. Los NMS proporcionan la carga para el procesamiento y recursos de memoria requeridos para el manejo de red, uno o más NMSs deben existir en toda red gestionada.

También es importante resaltar que SNMP utiliza el protocolo de comunicación UDP [Sta93] por lo tanto no garantiza confiabilidad en la entrega de los mensajes. Esto no es necesariamente una desventaja en el caso de este proyecto, ya que el envío de información por canales inalámbricos tampoco brinda garantía en la entrega de datos.

#### *5.1.1 Comandos Básicos de SNMP*

Los procesos de captura de información que realiza SNMP, tanto de la red como un todo, como de cada uno de los dispositivos que la componen, son llevados a cabo a través de una serie de cuatro comandos básicos, que al ser combinados permiten satisfacer los requerimientos de información de los usuarios, estos comandos son: *get*, *set*, *response* y *trap*.

El comando *get,* [Sta93] también llamado *read* por CISCO [Cis09], es usado por un NMS para solicitar información de uno de los dispositivos manejados en la red. Mediante este comando el NMS puede examinar diferentes valores de los parámetros de operación que son mantenidos por los dispositivos manejados.

El comando *set*, [Sta93] también llamado *write* por CISCO [Cis09], es usado por un NMS para controlar dispositivos manejados. Mediante este comando el NMS está facultado para cambiar el valor de los parámetros de operación almacenados en los dispositivos manejados.

El comando *trap* [sta93] es utilizado por los dispositivos manejados para reportar la ocurrencia de eventos asíncronos a los NMS, cuando cierto tipo de eventos ocurren el dispositivo manejado puede enviar una notificación al enviar un *trap* al NMS, sin tener que haber recibido previamente un *get*.

#### *5.1.2 MIB: Management Information Base*

Una *Management Information Base* (MIB) es una estructura, que está organizada de manera jerárquica, y que almacena una colección de información referente a datos, configuración, estado u operación de un equipo de cómputo o comunicaciones. Una MIB está compuesta por un conjunto de identificadores de objetos (o ID de objetos) que diferencian y describen cada una de las característica específica del dispositivo manejado. Las MIBs pueden ser accedidas utilizando un protocolo de gestión de redes como SNMP.

Cisco **Systems Inc.** [Cis09] explica que existen dos tipos de objetos manejados: escalares y tabuladores. Un objeto escalar define una instancia de un objeto. Un objeto tabular define múltiples instancias de objetos relacionados que están agrupados en tablas MIB.

Stallings [Sta93] indica que para que una MIB pueda cumplir las necesidades de un sistema de manejo de redes,

esta debe cumplir con dos objetivos: el objeto o los objetos utilizados para representar un recurso deben ser los mismos en cada nodo y debe existir un esquema de representación común para soportar la interoperabilidad.

### *5.1.3 Aplicaciones Proxy Forwarder*

Una aplicación *proxy forwarder* [LMS02] es un programa de software que puede reenviar mensajes SNMP a otros componentes.

Se propone el uso de un *proxy forwarder* SNMP, tal como se observa en la figura 2, debido a que los dispositivos de una RIS son limitados en capacidad de almacenamiento y memoria como para ejecutar una interfaz completa de manejo de red, también debido a que los protocolos usados en Zigbee no son compatibles con TCP/IP. Un *proxy forwarder* ha sido implementado en este trabajo para realizar las siguientes tareas:

- Ofrecer una interfaz para conocer el *status* de diferentes dispositivos de la RIS.
- Servir como repositorio de datos SNMP recolectados en la RIS.
- Integrar datos de diferentes dispositivos de la RIS para elaborar información de toda la red.
- Operar con protocolos TCP/IP. Si la información de la RIS va a ser utilizada fuera del alcance de la misma, éste funcionará como un *Gateway* para traducir la información entre los diferentes protocolos dentro de la RIS (ZigBee) y fuera de la RIS (TCP/IP).

El nodo *sink* de las RIS será el encargado de hacer las labores de *proxy forwarder*, éste debe estar conectado directamente al computador al cual se enviará toda la información recibida de la RIS.

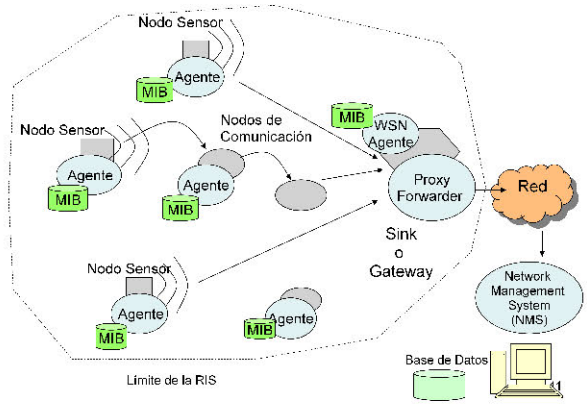


**Figura 2:** *Arquitectura general de Nirvana*

## 5.2 MIB desarrollada

La MIB general asociada a toda la red estará almacenada en el computador, el cual está conectado al nodo *sink de la RIS*. En esta MIB se guarda información de una serie de atributos que describen la configuración y operación de cada nodo sensor perteneciente a la red. En el caso particular de Nirvana se manejan los siguientes atributos para cada nodo sensor de la red:

- Identificador del nodo
- Cantidad de voltaje presente en el nodo
- Temperatura interna del nodo

## 5.3 Cacti: Herramienta de visualización

Para visualizar los datos, de la MIB, recolectados por la aplicación se utiliza la herramienta Cacti [Cac10]. Se seleccionó esta herramienta por ser sencilla de utilizar, fácil de instalar, además había sido utilizada en un proyecto anterior en un escenario similar. Cacti es un *frontend* completo para RRDTool que guarda toda la información necesaria para crear gráficos y poblarlos con datos guardados en una base de datos MySQL. El *frontend* es manejado completamente a través de PHP. Además de ser capaz de mantener gráficos, fuentes de datos y archivos *round robin* en una base de datos, maneja la recolección de los datos. RRDtool [Rrd10] es un estándar de código abierto que implementa una estructura circular de almacenamiento de datos que permite el registro de datos de rendimiento y que ofrece un sistema gráfico para visualizar series de datos en el tiempo. *Round robin* se refiere a una estrategia de bases de datos con tamaño determinado donde los datos se guardan de manera circular, es decir, al llegar al final del espacio destinado a guardar datos se sobrescribe el principio.

Para realizar el manejo de datos, se proporciona a Cacti el *path* a un script de Perl externo cuyo objetivo es el de recolectar la información como un trabajo programado y poblar los archivos *round robin* de la herramienta RRDTool. Una vez que una o más fuentes de datos son definidas, un gráfico RRDTool puede ser creado utilizando los datos recopilados. Un área de selección de color y texto de relleno automático ayudan a hacer el proceso de creación de gráficos de manera más sencilla.

De esta manera la MIB de la aplicación estará reflejada en archivos *round robin*, por lo que su información podrá ser accedida por medio de Cacti de manera gráfica y también a partir de archivos de texto en formato csv, que nos muestran la información recopilada.

## 5.4 Estructura de los mensajes

El formato de los mensajes SNMP está compuesto por un *message header* y un PDU (*Protocol Data Unit* o Protocolo de Unidades de Datos). En la sección del *message header* están definidos los campos *community name* y SNMP *versión*, el campo *community name* sirve como un *password* para proporcionar seguridad, el campo SNMP *version* especifica la versión de SNMP que se está utilizando. Para este proyecto se trabajó con SNMPv1 que aunque contiene un conjunto más simple de operaciones, permite trabajar con los aspectos básicos de gestión de red y en nuestro caso permite el envío de los valores sensados por la RIS. Con el objetivo de ahorrar espacio en los nodos, ya que estos tienen una capacidad limitada, sólo se envía el PDU en los mensajes, después en el nodo *sink* se puede agregar el *header* , antes de que se vaya a enviar el mensaje SNMP a otro dispositivo fuera de la RIS.

En la figura 3 se muestra la estructura de los mensajes SNMPv1 para las operaciones *get*, *set* y *response*, existe una estructura diferente para la operación *trap* sin embargo, en este trabajo no fue implementada con el objetivo de que todos los mensajes que viajen por la red tengan el mismo formato.

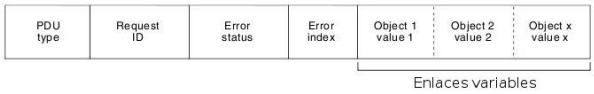


**Figura 3:** *Estructura general de un Mensaje SNMP*

Como se puede apreciar los mensajes en SNMP tienen una longitud no especificada, esto no es un problema en las redes tradicionales que emplean TCP/IP, ya que en este tipo de redes la segmentación permite dividir una información relativamente larga en paquetes o unidades más pequeñas y manejar el reensamblaje de las mismas antes de ser entregadas. En las RIS a pesar de que es posible implementar la fragmentación, no existen implementaciones estándares y programar esta característica puede complicar más allá de lo necesario el manejo de la red, por esto hemos decidido que cada mensaje SNMP de la RIS se coloque en un paquete de tamaño acotado y que sólo se trabaje con una variable a la vez. De esta manera los mensajes tendrán un tamaño apropiado y bien definido, se ha decidido acotar los mensajes de la red a un tamaño de 26 bytes. En cada uno de ellos hay valores referentes a exactamente un objeto manejado de la MIB, es decir uno de los atributos de un nodo.

A partir de la estructura de los mensajes SNMPv1 para las operaciones *get*, *set* y *response*, que se puede observar en la figura 3, se ha definido la estructura de datos SnmpMsg asociada a los mensajes de la red.

A continuación se explica cómo están formados los mensajes que viajan por la red y cuál es la utilidad de cada uno de los campos que lo conforman.

La trama de los mensajes SnmpMsg que emplea Nirvana, está conformado por nueve campos, todos de tipo de dato nx_uint16_t, entero sin signo de 16 bits proporcionado por TinyOS, tal como se muestra en la figura 4, la explicación de que datos deben ir en cada uno de estos campos se brinda a continuación:

- sourceNodeid: indica el identificador del nodo sensor que ha generado el mensaje.
- destinationNodeid: indica el identificador del nodo destino, el nodo *sink*.
- nodeidSender: indica el identificador del último nodo que ha enviado el mensaje. Si es la primera vez que se envía el mensaje este campo coincide con el sourceNodeid, si el mensaje está siendo retransmitido por algún otro nodo en su camino al nodo *sink*, entonces este campo tendrá el identificador de dicho nodo.
- object: hace referencia a un objeto de la MIB, relacionado a un atributo de los nodos de la RIS.
- value: es el valor del objeto en cuestión.
- pduType: especifica el tipo de PDU transmitido: *get, set* o *trap*.
- requestId: se utiliza para asociar los SNMP *request* con sus respuestas correspondientes.
- errorStatus: indica alguno de una serie de errores y tipos de errores, este campo sólo es establecido por operaciones de respuesta, cualquier otra operación asigna cero a este campo.
- errorIndex: asocia un error con una instancia particular de un objeto, este campo sólo es establecido por operaciones de respuesta, cualquier otra operación asigna cero a este campo.

Como se puede notar, los último seis atributos del tipo de datos SnmpMsg tienen los mismos nombres que los presentados en la figura 3. Se desea que los mensajes que viajan en la RIS sean lo más parecido posible a un mensaje SNMP original. Adicionalmente se agregan los tres primeros campos que son información de encaminamiento del mensaje, muy parecido a los mensajes SNMP que son encapsulados en el protocolo de comunicación UDP/IP.

| Source Nodeid | Destination Nodeid | Sender nodeid | object Type | value | pdu | requestId | errorStatus | errorIndex |
|---|---|---|---|---|---|---|---|---|

**Figura 4:** *Estructura general de un Mensaje SnmpMsg de Nirvana*

### 5.5 Descripción de la Operación de los Nodos de Nirvana

En esta aplicación tenemos tres tipos de nodos:

- Nodos sensores: son nodos que estarán realizando mediciones periódicas, enviando la información recolectada a través de la red, el objetivo es que esta información llegue hasta el nodo *sink*.
- Nodos enrutadores: su misión es enrutar o encaminar la información enviada por los nodos sensores para que llegue efectiva y correctamente a su destino, el nodo *sink*. En el caso de este primer prototipo se utilizará el algoritmo de enrutamiento por inundación o *flooding*.
- Nodo *sink*: Es el nodo que se encarga de recolectar toda la información enviada por los nodos sensores, además está encargado de realizar la labor de *proxy forwarder*.

Los nodos sensores realizan mediciones de los valores de interés de manera continua, con la información obtenida forman mensajes de tipo SnmpMsg y los envían hasta el nodo *sink* por medio de la RIS. A través del computador conectado al nodo *sink* el usuario tiene acceso a los datos por medio de Cacti, la herramienta de visualización. En la figura 5 se puede observar el diagrama de despliegue donde se muestran cada uno de los diferentes componentes de la arquitectura de Nirvana.

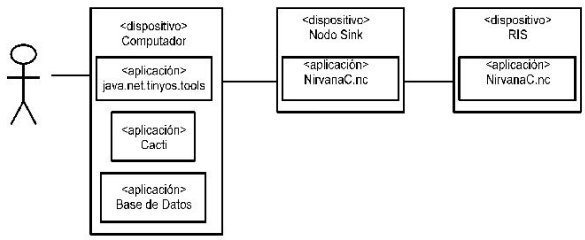


**Figura 5:** *Diagrama de despliegue de la aplicación Nirvana*

Como se mencionó anteriormente se enviará un mensaje por cada medición realizada, en el caso de que un nodo sensor esté realizando mediciones de más de un parámetro, como es el caso del ejemplo desarrollado, después de enviar un mensaje se debe esperar un tiempo prudencial antes de enviar el siguiente, con el objetivo de que el nodo receptor tenga tiempo de procesar cada mensaje correctamente.

Podemos ver los mensajes generados por los nodos sensores y enviados al nodo *sink* como operaciones tipo *trap*

debido a que son generados por los nodos sensores, como mencionamos anteriormente la estructura específica de los mensajes tipo *trap* no ha sido implementada, es por ello que estos mensajes siguen la misma estructura de un mensaje *get respons,e* aun cuando no se haya realizado un *get request* anteriormente. Stallings [Sta93] explica una estrategia llamada *Trap – Directed Polling*, la cual es muy parecida a la que se está utilizando, esta estrategia está basad en el hecho de que en el período de inicialización, y en intervalos poco frecuentes, se realicen operaciones tipo *get*, para establecer los parámetros básicos a ser consultados, una vez cumplido esto, se deja de realizar el *polling* y se permite que cada uno de los dispositivos manejados de la red, en el caso de Nirvana los nodos sensores, sean los que notifiquen por su cuenta los eventos que registran.

También se ha desarrollado una versión de la aplicación con enfoque de operaciones *get*, donde cada cierto tiempo el nodo *sink* realiza un *get request* a los nodos sensores y estos responden con los valores recolectados. El nodo *sink* no enviará una siguiente solicitud, *get request*, hasta que la anterior haya sido respondida, este paradigma de comunicación es conocido como *stop and wait*.

En función de lo diseñado, implementado y explicado contamos con una aplicación basada en SNMP que puede ser utilizada para capturar cualquier parámetro de interés en el ambiente industrial mediante una RIS, siempre que los nodos sensores tengan la capacidad de medir dicho parámetro.

La aplicación ha sido desarrollada en nesC y probada para los nodos tmote invent y tmote sky, utilizando el sistema operativo TinyOS. Moteiv [Mot04] nos explica que estos nodos sensores poseen un sensor de temperatura y humedad manufacturado por Sensirion AG [Sen10] por medio de los cuales pueden sensar estos valores, para después formar los mensajes que se envían al nodo *sink*.

## 6. Conclusiones y Recomendaciones

En este trabajo se diseñó y desarrolló una aplicación para recolectar datos de sistemas físicos de forma continua, ubicando nodos sensores en puntos de interés. Ésta ha sido desarrollada con la posibilidad de modificar fácilmente los parámetros sensados de forma tal que se pueda adaptar a nuevas necesidades de información de forma rápida y eficiente.

En Nirvana se logró desarrollar una aplicación para recolectar datos, que unida a una herramienta de visualización como Cacti, permite que la información recolectada por la RIS pueda ser accedida por los usuarios. Para realizar dicha integración primero se realizó un estudio del protocolo SNMP y después se desarrolló un protocolo muy similar para la recolección y el manejo de los datos recolectados en los nodos sensores, la similitud de los procesos de recolección de información de las herramientas de gestión de redes, con la recolección de datos sensador por nodos de una red inalámbrica de sensores, permitio la rápida integración de una herramienta de visualización como Cacti, de forma tal que los usuarios puedan consultar, a través de esta herramienta, los datos actuales e históricos, que la red inalámbrica de sensores ha estado recolectando.

## Referencias


[ASSC02] Akyildiz, I.F., Su W., Sankarasubramaniam Y., Cayirci E.: Wireless Sensor Networks: A Survey. Computer Networks, 38(4): pp 393-422, (March 2002)

[BDZ07] Berruti L., Denegri L., Zappatore S.: Wireless Sensor Networks and SNMP: Data Publication Over and IP Network. Wireless Communications 2007 CNIT Thyrrenian Symposium Signals and Communication Technology, 2007, 5, 295-308. http://www.springerlink.com/content/v78114237084t53m/

[Cac10] Cacti.: (2010). Recuperado el 17 de julio, del 2010 http://www.cacti.net/

[Cis09] Cisco Systems Inc.: Internetworking Technologies Handbook. (2009). http://docwiki.cisco.com/wiki/Simple_Network_Management_Protocol

[CS91] Comer D., Stevens, D.: Internetworking with TCP/IP. Volumen I: principles, protocols and architecture. Prentice-Hall international Editions. Second Editions. 1991.

[DG10] De Llano L., Gonzalez R.: Una Herramienta para el Despliegue de una Red Inalámbrica de Sensores. Jornadas de Investigación de la Facultad de Ingeniería JIFI 2010 y el Encuentro Académico Industrial.. Caracas, Venezuela. Noviembre 2010

[HKD06] Hongseok J., Kugsang J., Deokjai C.: Delivery and Storage Architecture for sensing information using SNMP. Proc. IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.4, April 2006. pp 130-134

[Lan08] Landaeta M.: Interfaz de monitoreo para una red inalámbrica de sensores (WSNMonitor). Trabajo Especial de Grado. Universidad Simón Bolívar. 2008.

[LMK*08] Lim Y., Messina M., Kargl F., Ganguli L., Fischer M., and Tsang T.:SNMP-Proxy for Wireless Sensor Network. Proc. of the Fifth international Conference on Information Technology, Singapore, 29 August 2008.

[LMS02] Levi D., Meyer P., and Stewart B.: Simple Network Management Protocol (SNMP) Applications, STD 62, RFC 3413, (December 2002). http://trac.tools.ietf.org/html/rfc3413

[MSK*04] Minami M., Saruwatari, S., Kashima, T., Morito, T., Morikawa, H., Aoyama, T.: Implementation-based approach for designing practical sensor network systems Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC'04) - Volume 00. (30 Nov.-3 Dec. 2004), pp 703- 710.

[MUVW07] Miorandi, D., Uhlemann, E., Vitturi, S., Willig, A.: Guest Editorial Special Section on

Wireless Technologies in Factory and Industrial Automation – Part I . Industrial Informatics, IEEE Transactions. (2007). pp 1-2.

[Mot04] Moteiv Corporation: Tmote Sky : Datasheet. p 4 (2004)

[Nes10] NesC (2004). Recuperado el 18 de julio de 2010, de http://nescc.sourceforge.net/.

[Rrd10] RRDtool (2010). Recuperado el 17 de julio, de http://oss.oetiker.ch/rrdtool/

[Sen10] Sensirion (2010). Recuperado el 17 de septiembre, de http://www.sensirion.com/

[Sta93] Stallings W. (1993). SNMP, SNMPv2, and CMIP The Practical Guide to Network – Management Standards. pp 72, 75.

[Tin10] TinyOS (2010). Recuperado el 18 de julio de 2010, de http://www.tinyos.net/.