# Arquitectura de descubrimiento de servicios para entornos hospitalarios (módulo tiny application)

***José Luis Santiago López***
Universidad de la Sierra Sur
*jlsantiagolopez@hotmail.com*

***Víctor Alberto Gómez Pérez***
Universidad de la Sierra Sur
*vgomez@unsis.edu.mx*

***Adrián Josué Ramírez Díaz***
Universidad de la Sierra Sur
*adrianejutla@gmail.com*

***Alejandro Jarillo Silva***
Universidad de la Sierra Sur
*ajarillo@unsis.edu.mx*

***Juan Carlos Santiago López***
Universidad de la Sierra Sur
*jcarloosstg@gmail.com*

## Resumen

El cómputo ubicuo está dando origen al diseño y a la implementación de sistemas inteligentes que se adaptan al usuario y que se utilizan de manera intuitiva. Un entorno ubicuo consta de cientos de computadoras colocadas en todas partes dentro de un

espacio físico, todas ellas unidas por redes de alta velocidad y capaces de detectar personas, objetos y dispositivos.

Generalmente, el entorno de trabajo de los hospitales está orientado al trabajo en equipo ya que, de una manera constante, se lleva a cabo la colaboración entre los diferentes campos de experiencia de los trabajadores. En este artículo se toman en cuenta los aspectos mencionados anteriormente para la implementación del cómputo ubicuo en entornos hospitalarios, creando una arquitectura y un sistema que lleva por nombre Tiny Application que tiene como objetivo permite a usuarios interactuar y controlar servicios por medio de un dispositivo móvil.

**Palabra(s) Clave(s):** cómputo ubicuo, entornos hospitalarios, sistemas de descubrimiento.

## 1. Introducción

En los últimos años, hemos sido testigos de importantes cambios derivados de la utilización de la tecnología de las telecomunicaciones (e.g., el creciente acceso a redes de área amplia mediante conexiones cableadas/inalámbricas o telefonía móvil) para satisfacer algunas necesidades de asistencia social (e.g., asistencia de salud domiciliaria a distancia) y de comunicación (e.g., transmisión de mensajes multimedia). Los beneficios de esta tecnología también se hacen evidentes en entornos laborales, donde los usuarios pueden acceder fácilmente a diversos servicios (e.g., consulta a bases de datos, navegación en la Web e impresión). En tales entornos, se puede asumir que la conectividad proporcionada por las redes corporativas es fiable, continua y ofrece un alto ancho de banda.

Sin embargo, debido al creciente uso de dispositivos móviles (e.g., smartphones, computadoras portátiles y PDAs) para interactuar con los servicios disponibles en el entorno, las personas tienden a volverse nómadas. A medida que se desplazan, las personas pueden encontrar cambios sustanciales referentes: 1) a la forma de

interacción con los dispositivos móviles, 2) a los tipos de servicios disponibles, 3) al medio de comunicación y 4) al ancho de banda. Los dispositivos móviles no sólo se vuelven más pequeños, baratos y poderosos sino también ejecutan más aplicaciones y servicios de red. Estas características incrementan el uso del cómputo móvil y ubicuo.

El avance tecnológico del cómputo móvil ha conducido a un cambio revolucionario en la sociedad. Estamos pasando de la era de la computadora personal (i.e., un dispositivo de cómputo por persona) a la era del cómputo ubicuo en la que un usuario emplea simultáneamente varios dispositivos de cómputo heterogéneos para acceder, en cualquier momento, a la información requerida [2]. El cómputo móvil y el cómputo ubicuo representan pasos evolutivos importantes en la línea de investigación de los sistemas distribuidos, la cual se remonta a la década de 1970.

Hoy en día, la idea de la computadora personal está quedando poco a poco fuera de lugar debido a que los dispositivos de cómputo emergentes tales como laptops y smartphones, solo son un paso transitorio para lograr un aprovechamiento real del potencial intrínseco de las tecnologías de la información [1]. Se pretende que tales dispositivos realicen un procesamiento integral y transparente a los usuarios, tomando en cuenta el entorno físico de los usuarios, en primer plano, y la capacidad de procesamiento de las computadoras, en segundo plano.

El gran ideal de Mark Weiser, considerado como el padre del cómputo ubicuo, era que la computadora se convirtiera en un artefacto tan incorporado, adaptable y natural, que se utilizara sin siquiera pensar en ella. La idea principal de Weiser consiste en la desaparición física de la computadora y en la incorporación de capacidades de conmutación y comunicación en la mayoría de los objetos físicos cotidianos, con el fin de formar una gran red de dispositivos interconectados. De esta manera, surge el tercer paradigma de las ciencias de la computación, el ya mencionado cómputo ubicuo, cuyo objetivo es crear entornos físicos inteligentes mediante el uso de sistemas computacionales, que estén disponibles pero que se vuelvan invisibles a los usuarios.

Los sistemas creados para este tipo de entornos se centran principalmente en la administración de información, en el soporte de actividades y en la colaboración entre trabajadores. Estos sistemas permiten organizar los servicios y datos de los trabajadores en términos de actividades, i.e., el entorno deberá ser capaz de determinar qué actividad le corresponde llevar a cabo y con qué servicios puede contar cada trabajador, dependiendo del lugar donde se encuentre.

Las clínicas y los hospitales manejan grandes volúmenes de datos compartidos, tales como registros de pacientes y rayos X. El trabajo que se presenta en este entorno está orientado a la colaboración entre personas, ya que las actividades implican hablar con los pacientes, atender conferencias e intercambiar opiniones entre colegas. Los médicos y las enfermeras usualmente no tienen una estación de trabajo fija, ya que la naturaleza de su trabajo hace que estén en constante movimiento.

La tecnología requerida para diseñar sistemas ubicuos se presenta en tres partes: 1) dispositivos de bajo costo; 2) computadoras con poco poder de procesamiento pero con grandes despliegues y software para soportar sistemas ubicuos; y 3) una red que vincule todos estos componentes. Las tendencias actuales sugieren que los dos primeros requisitos se están cumpliendo [14].

En este artículo se presenta la implementación del sistema Tiny Application que permite a usuarios nómadas interactuar y controlar servicios en un hospital a través de un dispositivo móvil. El sistema Tiny Application es parte de una arquitectura para descubrimiento de servicios que también es descrita en este artículo. En la sección dos se presentará el estado del arte, posteriormente se hará el planteamiento del problema en la sección número tres. Desde la sección cuatro a la siete se mostrará el diseño y la implementación del sistema, en la sección ocho se presentarán los resultados obtenidos hasta el momento, en la sección nueve y diez se presentan las conclusiones y referencias respectivamente, y finalmente en la sección once se presenta una breve reseña de los autores de la presente investigación.

## 2. Estado del arte

Debido al dinamismo de los entornos de cómputo ubicuo y a la creciente cantidad y diversidad de dispositivos móviles, es necesario proporcionar un mecanismo de descubrimiento y anuncio de servicios [10]. Algunas soluciones que ofrecen dicho mecanismo están ligadas a un protocolo de red (e.g., IP), mientras que otras forman parte de la arquitectura de un framework de desarrollo de servicios en entornos dinámicos [6]. Por lo tanto, la utilización de estas soluciones está muy ligada a la forma en que se desarrollan y se proporcionan estos servicios [3].

En la Fig. 1 se presenta una descripción de los sistemas de descubrimiento de servicio que se analizaron en la literatura. Se incluyen las siguientes características: a) el nombre del sistema de descubrimiento de servicios; b) si depende o no de un lenguaje; c) si está basado o no en un directorio; d) el tipo de almacenamiento de la información sobre los servicios disponibles; e) la técnica de descubrimiento de servicios utilizada por el sistema; f) la topología empleada para manejar la información de los servicios y de las consultas de los clientes; y g) el tipo de red.

Todos los sistemas analizados se basan en la comparación de atributos típicos (e.g., el tipo de servicio especificado por una URL) o de atributos de la interfaz de comunicación (e.g., la dirección IP y el número de puerto del host de servicio) para comprobar la existencia y la disponibilidad de los servicios requeridos por un usuario.

A excepción de Jini [13], todos los demás sistemas analizados son independientes del lenguaje. Dado que Jini es totalmente dependiente de Java, es necesario que exista una Máquina Virtual de Java en los dispositivos implicados.

Los sistemas de descubrimiento están o no basados en directorio para almacenar la información de los servicios disponibles en la red. Particularmente SLP [16], Ninja SSDS [7], Jini [13], Bonjour [15] y Salutation [8] están basados en un directorio, el cual recibe distintos nombres dependiendo del sistema, e.g., en SLP se le denomina

Dictionary Agent (DA). Para acceder a un servicio a través de un directorio, el cliente primero contacta el directorio central para obtener la descripción del servicio y posteriormente interactuar con el proveedor de servicios. Por el contrario, los sistemas UpnP [9] y DEAPSpace [12] no contienen un directorio para almacenar información de los servicios disponibles en la red. Los proveedores no distribuyen la descripción de sus servicios al resto de los nodos de la red sino que la mantienen en el dispositivo mediante el empleo de un caché local.

| Sistema | Dependiente de lenguaje | Basado en directorio | Almacenamiento de la información de los servicios | Técnica de descubrimiento de servicios | Topología | Tipo de red |
|---|---|---|---|---|---|---|
| SLP | × | ✓ | *Dictionary Agent* (DA) | Repositorio centralizado de servicios | - Centralizado<br>- P2P | Subred |
| Ninja SSDS | × | ✓ | *Lookup Service* | Descripciones XML | Cliente-Servidor | Estructura jerárquica de servidores |
| Jini | ✓ | ✓ | *Lookup Service* | Repositorio centralizado de servicios | Centralizado | Subred |
| UPnP | × | × | Punto de control | Inundación de peticiones | P2P | Subred |
| Bonjour | × | ✓ | Directorio jerárquico | Repositorio centralizado de servicios | - Centralizado<br>- P2P | Intranet |
| DEAPSpace | × | × | Vista global | Inundación de peticiones | P2P | *Mobile ad hoc 1-Hop* |
| Salutation | × | ✓ | *Salutation Manager* | Repositorio centralizado de servicios | - Cliente-Servidor<br>- P2P | Cualquier red depende del transporte |

**Fig. 1. Tabla comparativa de los principales sistemas de descubrimiento de servicios.**

La técnica de descubrimiento de servicios utilizada por los sistemas basados en directorio emplea un repositorio centralizado de servicios, a excepción de Ninja SSDS que hace uso de descripciones XML. Por el contrario, los sistemas no basados en directorio envían peticiones (i.e., inundación de peticiones) a todos los nodos de la red, con el fin de encontrar la descripción del servicio solicitado.

La topología se refiere a cómo los sistemas manejan la información de los servicios y las consultas de los clientes. La topología puede ser de tipo P2P, cliente-

servidor o centralizada. UPnP y Jini emplean respectivamente topologías de tipo P2P y centralizada dentro de una subred. Por el contrario, Ninja SSDS maneja una topología de tipo cliente-servidor con base en una estructura jerárquica de servidores. Dependiendo del transporte, Salutation puede trabajar mediante una topología tipo cliente-servidor o P2P dentro de cualquier tipo de red.

Todos los sistemas analizados exploran distintos aspectos del descubrimiento de servicios en sistemas distribuidos [4]. Empero, no ofrecen una solución adaptable a los problemas derivados de los entornos colaborativos dinámicos, e.g., la detección automática de los servicios prestados por las diferentes áreas de una organización o la creación dinámica de redes ad hoc para dar soporte tanto a la interacción usuario-servicio como a la interacción usuario-usuario dentro de un entorno hospitalario ubicuo.

## 3. Planteamiento del problema

En los últimos años, diversas organizaciones han diseñado y desarrollado sistemas de descubrimiento de servicios. En el ámbito académico se puede mencionar Intentional Naming System (INS) [1] del Instituto Tecnológico de Massachusetts y Ninja Service Discovery Service (SDS) [7] de la Universidad de California en Berkeley. De igual manera, los principales proveedores de software ofrecen sistemas de descubrimiento de servicios “integrados” a sus actuales sistemas operativos, e.g., Jini Network Technology [13] de Sun Microsystem, Universal Plug and Play (UPnP) [9] de Microsoft y Bonjour [15] de Apple. Algunas otras organizaciones de diversa índole también han propuesto sistemas de descubrimiento, e.g., DEAPSpace [12] de IBM Research, Salutation [8] del Consorcio Salutation, Service Location Protocol (SLP) [16] de Internet Engineering Task Force y Bluetooth SDP [5] del Grupo de Interés Especial en Bluetooth.

Todos estos sistemas dan soporte a entornos ubicuos en términos de topología de red o ubicación [11] [17]. Cada uno aborda diferentes cuestiones, pero la mayoría están diseñados para ámbitos computacionales. Aunque estos sistemas, no se orientan al

cómputo ubicuo más allá de estos fines, los distintos enfoques de diseño proporcionan puntos de referencia útiles para el futuro diseño de sistemas de descubrimiento.

En consecuencia, se observa que:

> La mayoría de estos sistemas únicamente se han enfocado en dar soporte a la interacción entre servicios computacionales con el fin de realizar una tarea específica;
>
> La interacción de un usuario con los servicios ofrecidos por el entorno, así como la interacción entre usuarios inmersos en dicho entorno, no han sido contempladas.

## 4. Diseño de la arquitectura de descubrimiento de servicios

El objetivo de la presente sección es describir la arquitectura funcional del sistema de descubrimiento de servicios. Este sistema tiene el objetivo de proporcionar un soporte a los usuarios nómadas para interactuar con los servicios disponibles y con otros usuarios nómadas dentro de un entorno hospitalario ubicuo.

## Arquitectura funcional

El sistema ha sido diseñado con la finalidad de permitir a un usuario nómada, localizado en un área autónoma: a) colaborar con otros usuarios, e.g., producción cooperativa de información e b) interactuar con los servicios disponibles en dicha área, e.g., solicitud de información. Las interacciones del usuario nómada están dirigidas por un flujo de trabajo sensible al contexto (definido en términos de la ubicación, del rol y de los objetivos del usuario). Cada área administrada por el sistema ha sido concebida como una entidad que se autoadministra y que se apoya en el sistema RBAC-Soft para comunicarse y coordinarse con otras áreas.

La Fig. 2 muestra la arquitectura funcional del sistema. El módulo Location Detector (ver Fig. 2 ref. #1) determina las coordenadas actuales del usuario nómada dentro de un

edificio por medio de las dos técnicas mencionadas anteriormente: 1) un sistema que utiliza triangulación de señales WiFi llamado WiTrack y 2) un sistema de reconocimiento de caras. A continuación, el módulo Location Detector transmite estas coordenadas al sistema RBAC-Soft, a fin de definir el área autónoma correspondiente (ver Fig. 2 ref. #2). Dependiendo de la ubicación actual del usuario nómada, RBAC-Soft determina: a) el rol atribuido al usuario nómada y b) los servicios disponibles a este rol (ver Fig. 2 ref. #3).

Con base en esta información contextual, el sistema RBAC-Soft crea un flujo de trabajo (o busca uno ya definido) para guiar al usuario nómada a lograr sus objetivos (ver Fig. 2 ref. #4). Cada una de las tareas del flujo de trabajo está asociada a un conjunto de servicios que permite a los usuarios nómadas llevar a cabo dichas tareas. De esta manera, RBAC-Soft carga el flujo de trabajo correspondiente en el dispositivo móvil del usuario nómada (ver Fig. 2 ref. #5) y solicita al módulo Service Manager que cargue las aplicaciones cliente o peer para que el usuario nómada pueda acceder a los servicios requeridos (ver Fig. 2 ref. #6).

Tan pronto como el usuario nómada tiene acceso a los servicios, este puede seleccionar uno de ellos desde su dispositivo móvil. El conjunto de servicios disponibles en un momento dado está controlado por el correspondiente flujo de trabajo, e.g., el flujo de trabajo puede activar: a) un servicio, en el caso de una única tarea serializada o b) varios servicios, en el caso de tareas cuyo orden de ejecución es irrelevante.

De esta manera, el módulo Ad hoc Creator (ver Fig. 2 ref. #7) establece dinámicamente una red ad hoc: a) entre el módulo Tiny Application (que se ejecuta en el dispositivo móvil del usuario) y el módulo Service Manager o b) entre dos módulos Tiny Application que se ejecutan en los dispositivos móviles de los usuarios nómadas que van a interactuar (ver Fig. 2 ref. #8). Este último tipo de red ad hoc facilita tanto el intercambio directo de información (e.g., facturas o vales) como la colaboración entre usuarios por medio de una comunicación peer to peer, en vez de pasar por un servidor central.

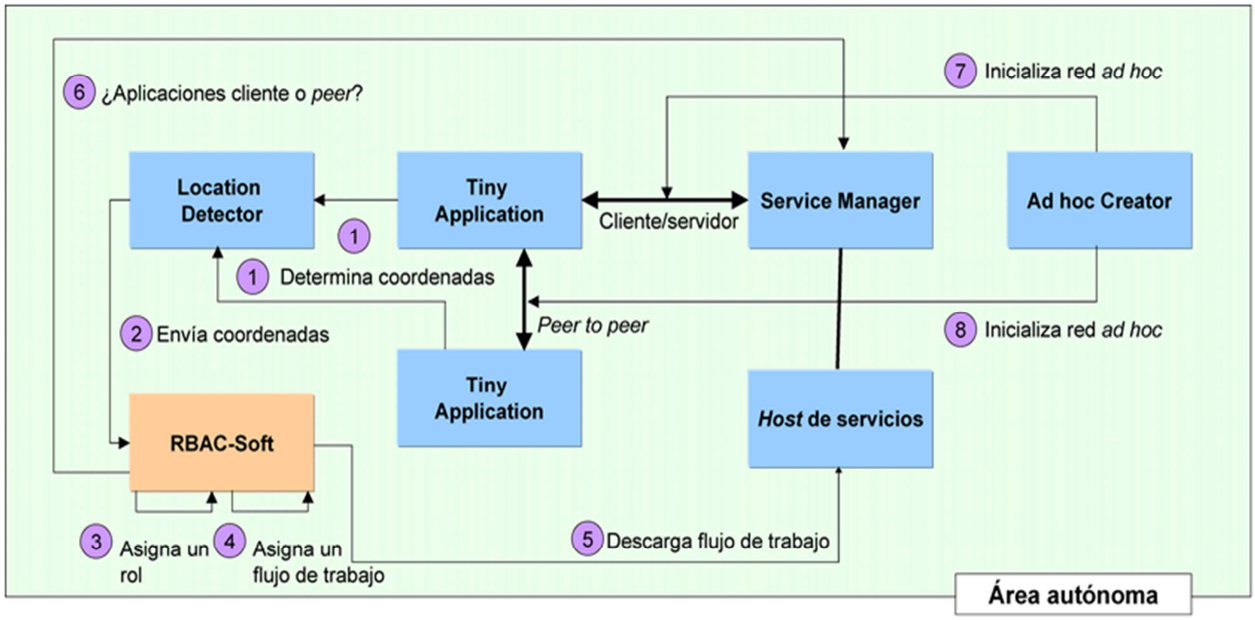


**Fig. 2. Arquitectura funcional del sistema de descubrimiento de servicios.**

## 5. Módulo TINY APPLICATION

Este módulo está embebido en el dispositivo móvil del usuario nómada. Se trata de una aplicación que ayuda al usuario nómada a interactuar con los servicios disponibles en el área autónoma actual o con otros usuarios nómadas por medio de una interfaz de usuario “amigable”.

Esta aplicación tiene tres funciones principales: 1) guiar al usuario nómada durante su recorrido por el hospital; 2) desplegar en la pantalla del dispositivo móvil los servicios ofrecidos por el área autónoma actual y 3) ofrecer las herramientas necesarias para permitir al usuario nómada interactuar con dichos servicios o con otros usuarios nómadas.

## 6. Desarrollo de la aplicación TINY APPLICATION

La aplicación Tiny Application consta de tres módulos: a) Módulo Servidor, b) Módulo Arduino Codificador Emisor y c) Módulo Receptor Decodificador.

## 6.1 Módulo servidor

El servidor consiste en un servidor web con soporte para PHP, en este caso se utiliza xampp, dentro del directorio de publicación (htdocs) se creará la carpeta móvil que contienen las páginas index.php, para autenticación de los usuarios (ver Fig. 3):

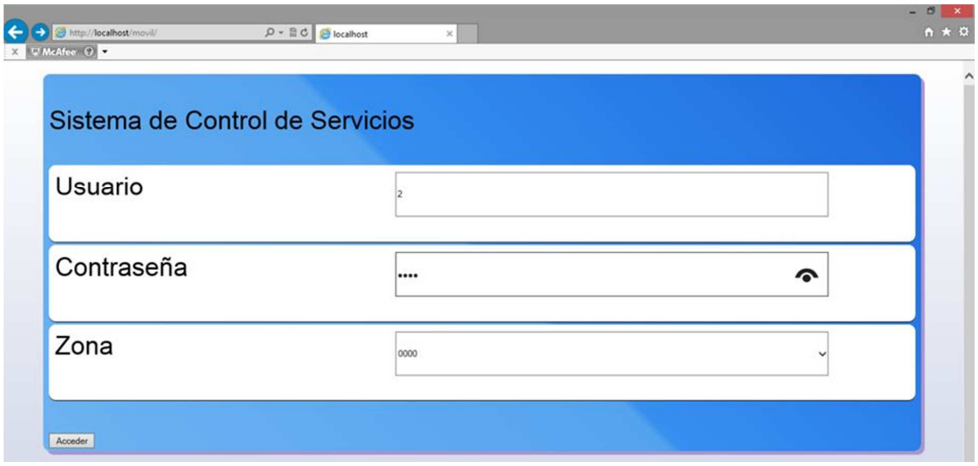


**Fig. 3. Autenticación en el sistema.**

Tras realizar una autenticación correcta mostrará la página dispositivos.php que permite indicar con que servicios cuenta el área en cuestión, ahí puede presionarse el botón para cambiar el estado, en este caso encender o apagar el foco (ver Fig. 4).

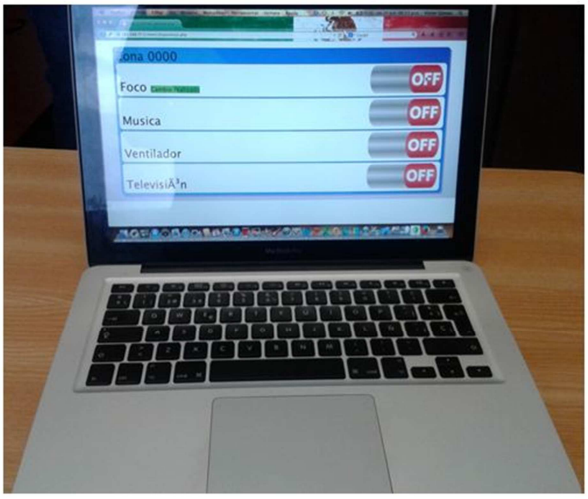


**Fig. 4. Listado de servicios.**

Adicionalmente es necesario un servidor de base de datos MYSQL con la BD ser_ubi que almacene la información de:

Dispositivos (ver Fig. 5) Emisora (ver Fig. 6):

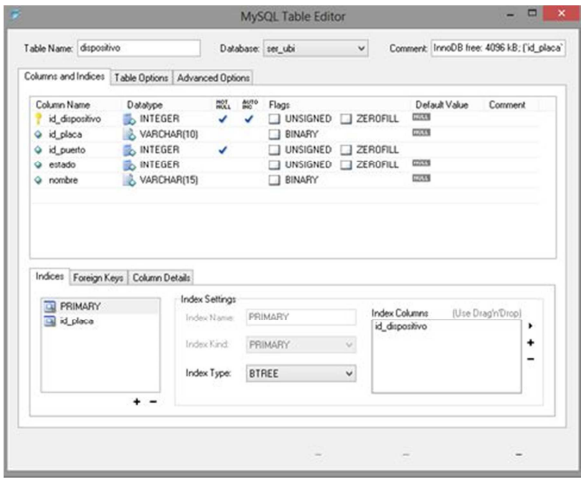


**Fig. 5. Tabla dispositivos.**

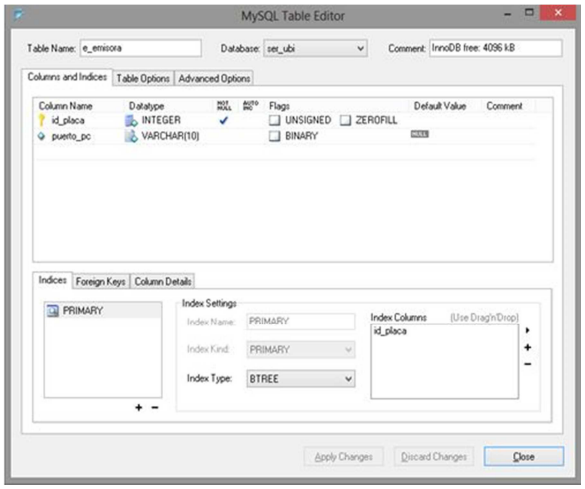


**Fig. 6. Tabla emisora.**

Receptora (ver Fig. 7) Usuarios – dispositivos (ver Fig. 8)

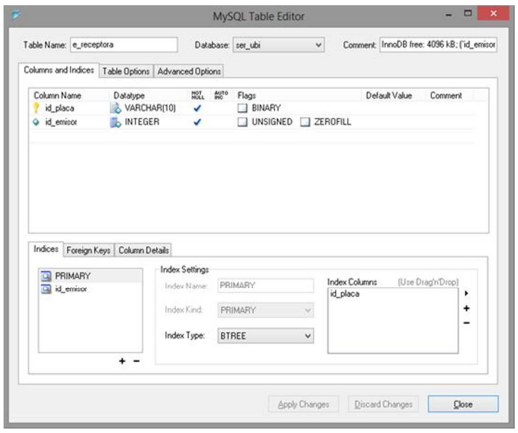


**Fig. 7. Tabla receptora.**

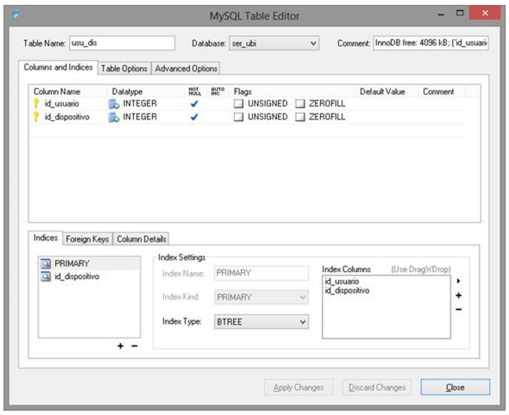


**Fig. 8. Tabla Usuarios-Dispositivos.**

Usuarios (ver Fig. 9):

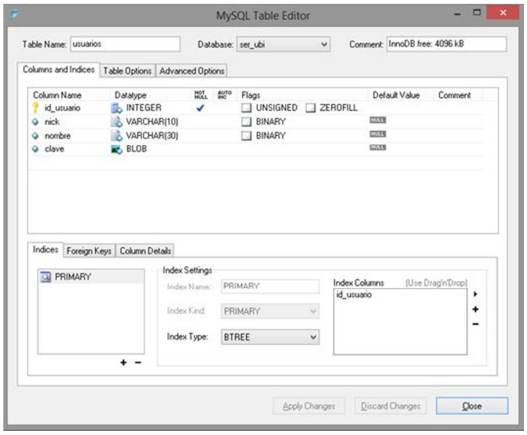


**Fig. 9. Tabla usuarios.**

## 6.2 Módulo Arduino codificador emisor

Este módulo (ver Fig. 10) se encarga de recibir desde el Módulo Servidor, una trama de doce Bits divididos en ocho Bits para saber el área en la que se encuentra el usuario y los otros cuatro Bits para identificar el servicio que se quiere activar. Posteriormente codifica esta trama y la envía por medio de radiofrecuencia al módulo Receptor Decodificador.

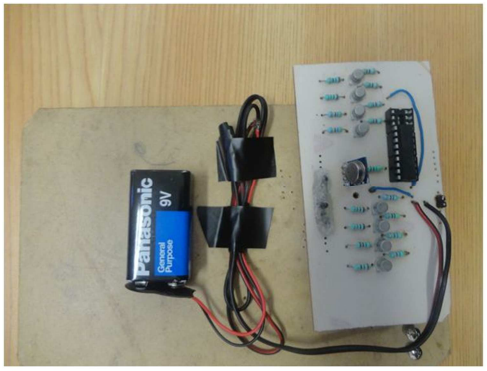


**Fig. 10. Módulo Arduino codificador emisor.**

## 6.3 Módulo receptor decodificador

Este módulo (ver Fig. 11) se encarga de recibir por radiofrecuencia la trama enviada por el Módulo Arduino Codificador Emisor, posteriormente decodifica esta trama (ver Fig. 12) identificando el área y el servicio seleccionado para posteriormente enviar una señal a un MOC y activar un Triac (ver Fig. 13) para cerrar el circuito apropiado para activar físicamente dicho servicio (ver Fig. 14).

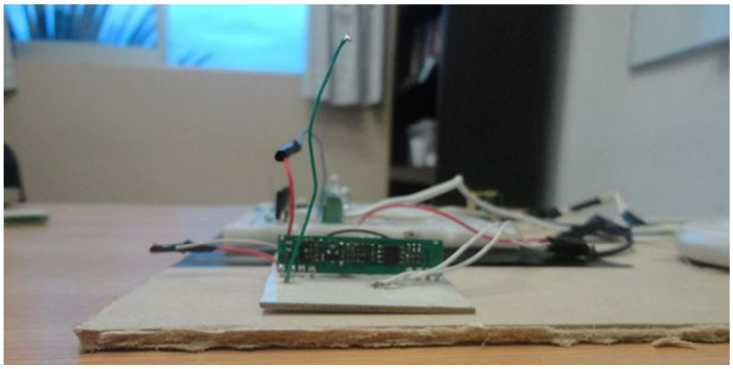

**Fig. 11. Módulo receptor decodificador – Vista de la tarjeta emisora. -**

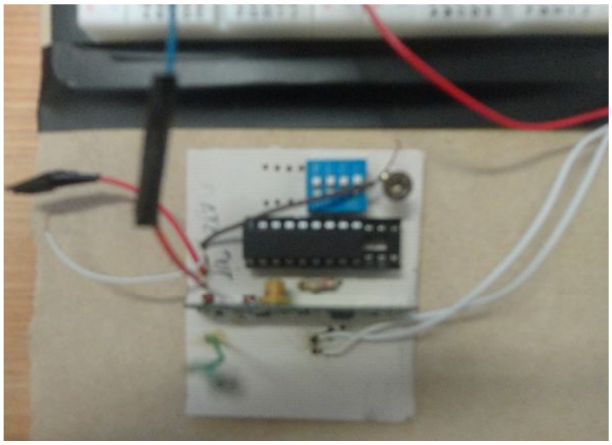

**Fig. 12. Módulo receptor decodificador – Vista superior.**

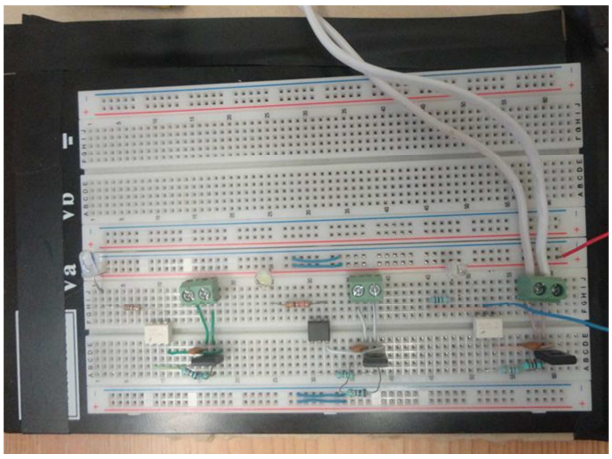

**Fig. 13. Módulo receptor decodificador – Vista de la fase de potencia.**

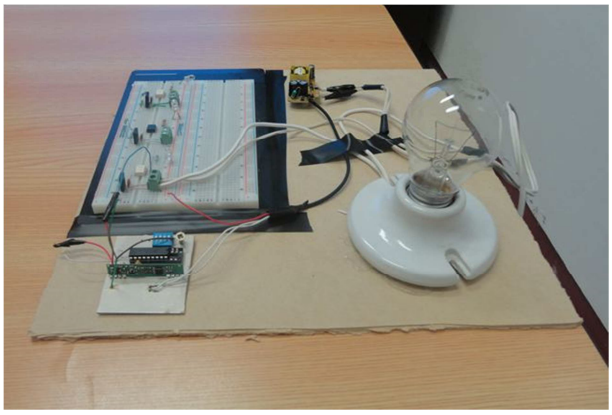

**Fig. 14. Módulo receptor decodificador completo.**

## 7. Puesta en marcha

A través de cualquier dispositivo con acceso a una red WiFi (e.g., teléfono celular, tableta, PC) podemos acceder al Módulo Servidor. Primeramente se pedirá "identificarse" al usuario, de esta manera se decidirá a qué servicios tiene derecho dependiendo del área en la que se encuentra. A continuación se le mostrará al usuario en su pantalla todos los servicios disponibles. El usuario podrá elegir entre activar o desactivar un servicio. En este ejemplo el usuario activará la luz de un foco (ver Fig. 15).


**Fig. 15. Vista de la interfaz de usuario desde un dispositivo iPhone 4.**

A manera de simulación se cuenta solamente con un área (00000000), desplegándose en ella cuatro servicios disponibles de los cuales uno de ellos esta funcional (el foco (0001)). Al elegir el servicio foco que se encuentra en el área 00000000, se construirá la trama 000000000001 y se enviará al Módulo Arduino Codificador Emisor el cual codificará esta trama y la enviará por medio de radio frecuencia al Módulo Receptor Codificador para que este decodifique la trama y decida qué servicio se activará (ver Fig. 16 y 17).

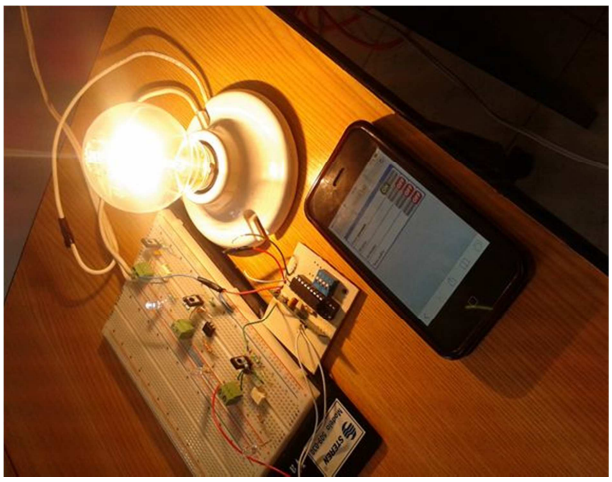

**Fig. 16. Servicio activado desde el iPhone.**

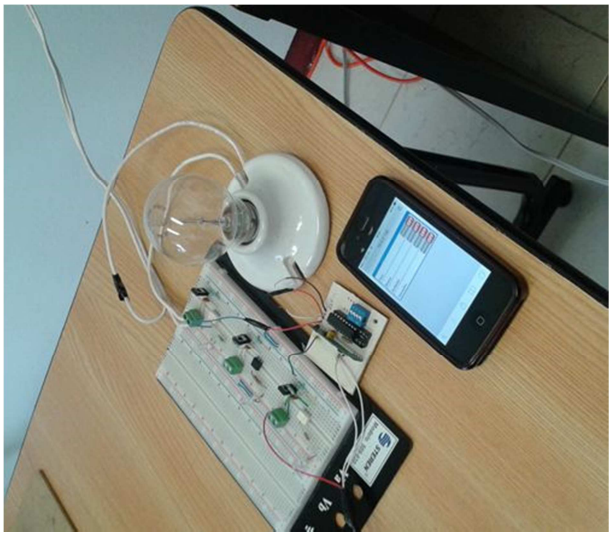

**Fig. 17. Servicio desactivado desde el iPhone.**

## 8. Resultados

Al implementar y poner en funcionamiento la aplicación Tiny Application se obtuvieron buenos resultados. Se pudo manipular de manera exitosa un servicio (el encendido y apagado de un foco). En esta primera versión se pueden controlar como máximo cuatro servicios, sin embargo, como trabajo futuro se puede extender este trabajo para manipular más áreas y más servicios inclusive por medio de comandos de voz.

Como futuras extensiones de esta arquitectura, se desarrollarán módulos para crear redes ad hoc por medio del protocolo ZigBee [18], el cual puede ser utilizado por aplicaciones que no requieran gran transmisión de datos. Además, el consumo de batería de un dispositivo ZigBee es mucho más bajo que el de los dispositivos WiFi. Asimismo, la arquitectura propuesta constituye la base para la definición de un framework que dará soporte a la colaboración nómada y ubicua.

Se contempla también la construcción de un mecanismo para detectar cambios contextuales en el área autónoma actual y en otras áreas, que puedan tener influencia en el plan de actividades de un usuario nómada y que dotarán de inteligencia a la arquitectura.

## 9. Conclusiones

En este artículo se propone una arquitectura para un sistema de descubrimiento de servicios para hospitales. El objetivo principal es facilitar la interacción entre los usuarios y los servicios proporcionados por las áreas autónomas, así como la colaboración entre los usuarios bajo contextos específicos.

En este artículo se presenta también el diseño de la aplicación Tiny Application, la cual es una aplicación que se accede a través de cualquier dispositivo con acceso a WiFi y permite manipular los servicios disponibles dentro de un área autónoma actual dependiendo del rol y la localización de los usuarios.

## 10. Referencias

[1] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, J. and Lilley, The Design and Implementation of an Intentional Naming System. In Proc. 17th ACM Symp. Operating System Principles (SOSP 99), December 1999. ACM Press, Charleston, South Carolina (USA). 186-201 pp.

[2] J. Bardram, “Pervasive Computing Support for Hospitals: An overview of the Activity-Based Computing Project”. IEEE Pervasive Computing. Vol. 6. No. 1. January 2007. 44-51 pp.

[3] C. F. A. Campo, D. Diaz, C. Garcia-Rubio, A. Lopez, “Secure Service Discovery based on Trust Management for ad-hoc Network. Universal Computer Science”. Vol. 12. No. 3. March 2006. 340-356 pp.

[4] G. Fortino, M. Lackovic, W. Russo, P. Trunfio, A discovery service for smart objects over an agent-based middleware. In Internet and Distributed Computing Systems. Springer Berlin Heidelberg. March 2006. 281-293 pp.

[5] Bluetooth and Bluetooth Smart. Washington University in Saint Louis, Lecture Note. 2014.

[6] W. Chen, I. Paik, P. C. Hung, “Constructing a global social service network for better quality of web service discovery. Services Computing”. IEEE Transactions on. Vol. 8. No. 2. 2015. 284-298 pp.

[7] S. Herborn, Y. Lopez, A. Seneviratne, A Distributed Scheme for Autonomous Service Composition. In Proc. of the First ACM Int. Workshop on Multimedia Service Composition. November 2005. ACM Press, Hilton. Singapore. 21-30 pp.

[8] A. Jamalipour, The Wireless Mobile Internet: Architectures, Protocols and Services. John Wiley and Sons, Inc., New York, NY (USA). March 2003.

[9] M. Jeronimo, J. Weast, UPnP Designe by Example: A Software Developer’s Guide to Universal Plug and Play. Intel Press. May 2003.

[10] S. Bradai, S. Khemakhem, M. Jmaiel, M. Discovering Services in Mobile Environments: Discussion and Evaluation of Trends. Handbook of Research on Architectural Trends in Service-Driven Computing. 2014. 299 pp.

[11] S. Cheshire, M. Krochmal, DNS-based service discovery. 2013.

[12] M. Nidd, Service Discovery in DEAPspace. IEEE Personal Comm. August 2001. 39-45 pp.

[13] S. Oaks, H. Wong, Jini in a Nutshell: A Desktop Quick Reference. O'Reilly and Associates, Inc., Sebastopol, CA, USA. March 2000.

[14] M. Weiser, The Computer for the 21st Century. Scientific American. IEEE Pervasive Computing. Sept. 1991. 19-25 pp.

[15] K. White, Apple Training Series Mac OS x Support Essentials. 2007. Peachpit Press. Berkeley, CA (USA).

[16] W. Zhao, H. Schulzrinne, "Enhancing Service Location Protocol for Efficiency, Scalability and Advanced Discovery". Journal of Systems and Software. Vol. 75. No. 1-2. February 2005. 193-204 pp.

[17] B. C. Villaverde, R. De Paz Alberola, A. J. Jara, S. Fedor, S. K. Das, D. Pesch, "Service discovery protocols for constrained machine-to-machine communications". Communications Surveys & Tutorials, IEEE. Vol. 16. No. 1. 2014. 41-60 pp.

[18] S. K. Chen, T. Kao, C. T. Chan, C. N. Huang, C. Y. Chiang, C. Y. Lai, P. C. Wang, "A reliable transmission protocol for ZigBee-based wireless patient monitoring. Information Technology in Biomedicine". IEEE. Vol. 16. No. 1. 2012. 6-16 pp.

## 11. Autores

José Luis Santiago López es estudiante de la Lic. en Informática de la Universidad de la Sierra Sur

Víctor Alberto Gómez Pérez actualmente es profesor investigador en la Universidad de la Sierra Sur adscrito al Instituto de Informática

Adrián Josué Ramírez Díaz es estudiante de la Lic. en Informática de la Universidad de la Sierra Sur

Alejandro Jarillo Silva actualmente es profesor investigador en la Universidad de la Sierra Sur adscrito al Instituto de Informática

Juan Carlos Santiago López es estudiante de la Lic. en Informática de la Universidad de la Sierra Sur