R. J. Caro, et al., "Middleware Seguro EP2P: un Desafío para las Redes Sociales", *XVIII Jornadas Telecom I+D*, 2008.
NICS Lab. Publications: https://www.nics.uma.es/publications

Middleware seguro EP2P: un desafío para las redes sociales

Rafael J. Caro Benito[1], Daniel Garrido Márquez[2], Pierre Plaza Tron[3], Rodrigo Román Castro[2], Nuria Sanz Martín[3], José Luis Serrano Martín[1]

Tecnatom[1], Universidad de Málaga[2], Telefónica Investigación y Desarrollo[3]

Correos Electrónicos: rcaro@tecnatom.es, dgarrido@lcc.uma.es, pierre@tid.es, roman@lcc.uma.es, nsanz@tid.es, jlserrano@tecnatom.es

***Abstract* — Los sistemas distribuidos en dispositivos embebidos representan un nuevo reto en el desarrollo de software. Estos sistemas han supuesto una importante revolución en el paradigma de la computación distribuida donde se intenta fragmentar un problema grande en múltiples problemas más pequeños. El nuevo escenario tiende entonces hacia sistemas en los cuales todos los elementos de la red se consideran iguales y los mecanismos de comunicación están basados en redes ad-hoc que se forman dinámicamente. De esta forma cualquier usuario de la red (en realidad cualquier elemento, hasta el más simple dispositivo) adquiere valor, a mayor colaboración, mayor éxito del sistema. Sin embargo, desde el punto de vista de la seguridad, estos sistemas son extremadamente vulnerables. En este artículo se presenta SMEPP, un middleware diseñado especialmente para sistemas P2P incluyendo aspectos de seguridad. SMEPP está diseñado para poder ser ejecutado en un amplio rango de dispositivos (desde redes de sensores hasta PC), y trata de facilitar el desarrollo de aplicaciones ocultando los detalles de la plataforma y otros aspectos tales como escalabilidad, adaptabilidad e interoperabilidad. Además el artículo presenta dos aplicaciones de alto nivel que utilizando este middleware pasan a ser más personales, más sociales y más baratas, haciendo que todos los usuarios de la red cobren mayor importancia.**

I. INTRODUCCIÓN

Al igual que los humanos captamos la información de nuestro entorno a través de los sentidos y utilizamos esa información para actuar de un modo u otro; los sistemas informáticos pueden captar la información a través de sensores, desplegados en forma de redes, y utilizarla para el tratamiento correspondiente. La información captada se puede utilizar para ofrecer a los usuarios servicios contextualizados y adaptados al entorno que les rodea en cada momento. Si unimos este concepto al paradigma de comunicación entre pares (P2P), donde no existen clientes ni servidores, sino que cualquier elemento de la red adquiere la misma importancia, nos encontramos frente al desafío de resolver sistemas de comunicación segura en dispositivos que no tienen por qué tener grandes capacidades de computación.

Una de las claves para el éxito de los sistemas distribuidos embebidos, conocidos como EP2P (*Embedded Peer-to-Peer*), pasa por facilitar la tarea de implementación de aplicaciones y servicios por medio de un middleware adecuado, que permita al diseñador y al programador abstraerse de problemas como no tener una red de infraestructura montada previamente o posibles vulnerabilidades en temas de seguridad; permitiendo reducir la complejidad de las aplicaciones y un abaratamiento de los costes. Este middleware oculta, por tanto, la complejidad de la infraestructura subyacente mientras proporciona interfaces abiertas para el desarrollo de aplicaciones a terceros.

El proyecto europeo SMEPP (*Secure Middleware for Embedded Peer-to-Peer system*) [1] trata de desarrollar un middleware específicamente pensado para ser desplegado en dispositivos limitados, que sea lo más eficiente posible en cuanto a consumo de energía, y que haga uso de algoritmos criptográficos y técnicas de prevención de ataques específicas para los problemas de seguridad más frecuentes en este tipo de sistemas distribuidos. En el proyecto además se valida dicho middleware usándolo en aplicaciones de alto nivel que se benefician de esa cooperación entre los usuarios; con estas aplicaciones se comprueba cómo un mismo servicio pasa a ser mucho más eficiente, seguro y social gracias al nuevo enfoque de colaboración entre los usuarios frente al clásico cliente/servidor.

El reto es trabajar con dispositivos de baja capacidad como los nodos de una red de sensores [2], que sólo disponen de una limitada capacidad computacional, lo que dificulta la inclusión de primitivas y servicios de seguridad, sin olvidar otro tipo de dispositivos con mayores capacidades como los PC. Además, el middleware desarrollado en SMEPP debe ser compatible con otros tipos de middleware existentes, de forma que puedan llegar a convivir en un mismo dispositivo.

Trabajos relacionados con el realizado en el proyecto SMEPP pueden encontrarse, por ejemplo, en RUNES [3], donde se desarrolla una arquitectura para sistemas empotrados en red capaz de funcionar en un amplio rango de dispositivos con posibilidad de configurar, desplegar o reconfigurar dinámicamente software para este tipo de sistemas. RUNES presenta un modelo de componentes para el desarrollo de su arquitectura proporcionando una API que permite abstraerse de los detalles del hardware del dispositivo utilizado. También MORE [4] proporciona una arquitectura basada en servicios de manera similar a SMEPP, funcionando en tres niveles de dispositivos que incluyen redes de sensores, PDA o PC. Como aspecto interesante, MORE incluye versiones simplificadas de algunos protocolos para funcionar en dispositivos con menor capacidad. Finalmente, el proyecto Amigo [5] está orientado a computación inteligente para el hogar, facilitando la

integración de servicios en todo tipo de dispositivos electrónicos utilizando mecanismos de descubrimiento. Todos estos proyectos fueron estudiados en SMEPP; sin embargo, ninguno de ellos trata a la vez todos los aspectos que SMEPP considera en cuanto a seguridad, utilización en redes peer-to-peer, servicios, etc.

II. ARQUITECTURA DEL MIDDLEWARE

La arquitectura del middleware desarrollada en SMEPP sigue un modelo P2P de componentes que proporciona las herramientas necesarias para adaptar el middleware a los diferentes dispositivos, aplicaciones y redes. No se trata entonces de desarrollar un middleware tan genérico que funcione en todo tipo de dispositivos de igual manera, sino de establecer un marco de componentes donde se pueda adaptar cada uno de esos componentes de forma eficiente según las capacidades de cada tipo de dispositivo. Se trata, por tanto, de hacer prevalecer la eficiencia y la seguridad, sin perder la compatibilidad entre dispositivos de distinta índole.

El carácter heterogéneo de los participantes en el proyecto SMEPP hace que el diseño de la arquitectura del middleware sea un desafío especial. Dos aspectos son esenciales en el diseño de la arquitectura: el manejo de los requisitos de seguridad en sus diferentes niveles y, que al mismo tiempo, la arquitectura sea suficientemente flexible, escalable y adaptable a los distintos tipos de dispositivos y plataformas donde SMEPP debe funcionar. El desarrollo de la arquitectura se ha realizado partiendo pues, de las opiniones de los diferentes expertos, teniendo en cuenta los requisitos de los diferentes escenarios e identificando los denominados *architectural drivers*, término que hace referencia a cuáles son los elementos, factores, requisitos, etc. que van a regir el diseño de la arquitectura. Concretamente, y partiendo de los diferentes requisitos encontrados, el diseño de la arquitectura del middleware está dirigido por la seguridad, adaptabilidad, escalabilidad, interoperabilidad con otro tipo de sistemas e implementación en plataformas heterogéneas.

La arquitectura está diseñada también pensando en su interacción con otro tipo de sistemas basados, por ejemplo, en OSGi [6] que pueden funcionar con pasarelas interconectando estos sistemas externos con el resto de la red P2P de SMEPP. De esta forma, es posible tener aplicaciones donde convivan, por ejemplo, el mencionado OSGi y SMEPP.

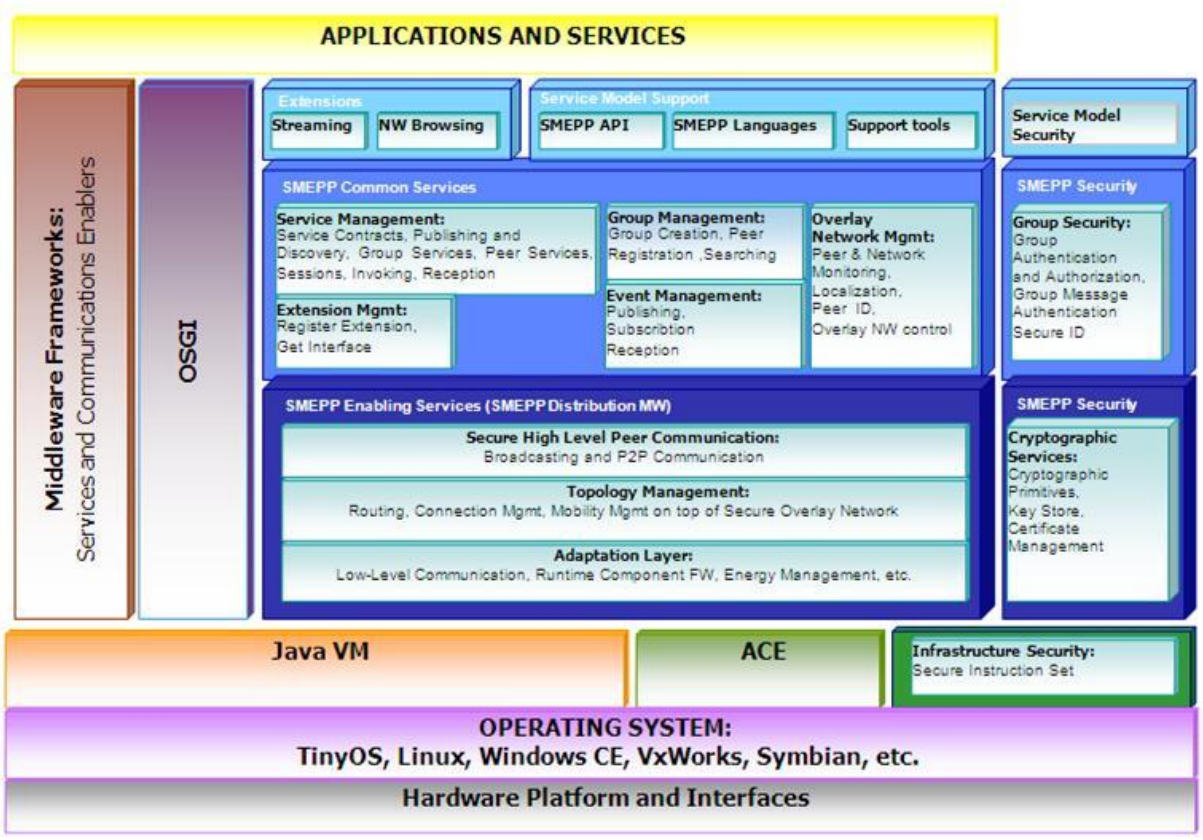

Figura 1. Arquitectura propuesta para el middleware SMEPP y su contexto

La arquitectura está dividida en tres marcos funcionales. En la capa más alta, SMEPP define un modelo de servicios abstracto, encima del cual se desarrollan las aplicaciones P2P. El papel principal del denominado *Service Model Support* es precisamente proporcionar al desarrollador de aplicaciones la API de SMEPP basada en las primitivas definidas por este modelo de servicios. En esta capa superior se encuentra también el soporte de extensiones de SMEPP que permite extender el

modelo de servicios así como las primitivas que este modelo proporciona, permitiendo adaptar de mejor manera el middleware a un dominio de aplicación en particular.

En un nivel intermedio se encuentran los denominados *SMEPP Common Services*, incluyendo los componentes necesarios para completar la funcionalidad del middleware, tales como la gestión de eventos, grupos, servicios y mensajes, monitorización del estado de los *peers*, notificando cuando se unen nuevos *peers* a la red o cuando alguno no responde, etc. Estos servicios comunes utilizan la tecnología basada en componentes, de tal forma, que cualquiera de estos componentes podría ser sustituido por distintas implementaciones siempre que se mantuvieran las interfaces, permitiendo así una fácil adaptación a distintos tipos de dispositivos o plataformas. A continuación se detallan los componentes de este nivel y sus responsabilidades:

- *Event Management*: proporciona maneras de crear, publicar, recibir o suscribirse a eventos en la red SMEPP.
- *Group Management*: permite gestionar los grupos de SMEPP. El concepto de grupos es una parte importante dentro de SMEPP, permitiendo formar agrupaciones de *peers* dentro de una red SMEPP. Este componente proporciona servicios para crear grupos, unirse o abandonar grupos, buscar o monitorizar grupos, etc. Todas estas operaciones requieren además la autenticación y autorización de los *peers*.
- *Service and Message Management*: proporciona los mecanismos para publicar servicios de SMEPP, descubrir servicios de *peers* o grupos y comenzar sesiones en los servicios. Una tarea importante de este componente es también gestionar los "contratos" de los servicios. Un contrato de servicio indica ciertas características que el servicio debe cumplir. Este componente debe además proporcionar mecanismos para el intercambio de mensajes entre *peers*, servicios o *peers* y servicios (denominados todos ellos *entidades*), implementando los protocolos de comunicaciones entre entidades con operaciones que permiten invocar operaciones y recibir respuestas.
- *Extension Management*: proporciona las herramientas y mecanismos que permiten integrar extensiones a SMEPP. Así por ejemplo, en dispositivos con mayores capacidades, podría proporcionar un entorno de ejecución para OSGi donde podrían insertarse o modificarse *plugins* en tiempo de ejecución.
- *Overlay Network Management*: este componente tiene la responsabilidad de permitir que los *peers* tengan *identidad*. La identidad de un peer es un concepto muy importante que debe ser tenido en cuenta para realizar todo tipo de operaciones. Este componente permite pues realizar operaciones, tales como registrar *peers* en la red SMEPP y que estos *peers* adquieran una identidad. Este componente puede ser también utilizado para monitorizar el estado de los *peers*, notificando cuando se unen nuevos *peers* a la red o cuando alguno no responde.

En un nivel inferior al de los componentes comentados se encontrarían los denominados *SMEPP Enabling Services*, que incluyen la comunicación básica peer a peer y los protocolos de comunicaciones subyacentes como, por ejemplo, los de enrutado seguro. Por último, y no menos importante, se incluye también un entorno de ejecución que permite la ejecución de todos los componentes de SMEPP que forman parte de su arquitectura. Los componentes de este nivel dependerán fuertemente de las restricciones de la infraestructura, plataforma de ejecución, dispositivo, etc. permitiendo lograr los objetivos mencionados de adaptabilidad y escalabilidad. Los tres componentes incluidos en el nivel inferior son los siguientes:

- *Secure High-Level Peer Communication*: proporciona al nivel superior las facilidades necesarias para la comunicación entre los peers de una forma segura y con un alto nivel de abstracción.
- *Secure Topology Management*: encargado de la seguridad en toda la red SMEPP, incluyendo conceptos como autenticación de nuevos *peers*, permisos para entrar en la red SMEPP y la protección de la información de enrutado que es intercambiada en la red.
- *Adaptation layer*: es el componente encargado de proporcionar una interfaz abstracta sobre la infraestructura de ejecución. Aísla los niveles superior de los detalles concretos de implementación ofreciendo unas primitivas para el envío de mensajes así como la gestión de los datos de energía (característica muy importante en dispositivos con pocos recursos). Es también responsable del soporte de ejecución del resto de componentes de SMEPP.

La seguridad es considerada también en la arquitectura, atravesando al resto de niveles e incluyendo componentes para la gestión de la seguridad en grupo o servicios criptográficos. Los componentes incluidos son los siguientes:

- *Group Security*: permite el mantenimiento de la seguridad en los grupos de SMEPP, siendo responsable de tareas como la autenticación de los *peers* que se quieren unir a un grupo.
- *Cryptographic Services*: proporciona la funcionalidad común al resto de componentes que requieren seguridad, incluyendo primitivas criptográficas (encriptar, desencriptar, firmas digitales, etc.).
- *Infrastructure Security*: utiliza las características específicas de algunos dispositivos en materia de seguridad. Por ejemplo, la utilización de instrucciones seguras que incrementan el procesamiento criptográfico.

III. ASPECTOS DE SEGURIDAD

La seguridad es un aspecto crucial para el funcionamiento adecuado de un sistema P2P basado en dispositivos embebidos, ya que estos sistemas se encuentran desprotegidos ante varios tipos de ataques externos e internos. Estos ataques permiten manipular los flujos de información, obteniendo y/o modificando los datos enviados dentro de la red. Todo esto es posible debido a las características limitadas de comunicación y capacidad en este tipo de redes P2P embebidas. Tal y como se ha mencionado anteriormente, uno de los objetivos primordiales del proyecto SMEPP es la integración de mecanismos de seguridad dentro de la arquitectura del middleware. Además, esos mecanismos de seguridad deben cumplir con una serie de requisitos para facilitar el uso del middleware: transparencia y adaptabilidad. Los desarrolladores de las aplicaciones deben disponer de una infraestructura segura y robusta ante ataques externos y/o internos de la forma más transparente posible. Además, esos mecanismos de seguridad deben poder adaptarse a las necesidades de seguridad del entorno y de la aplicación.

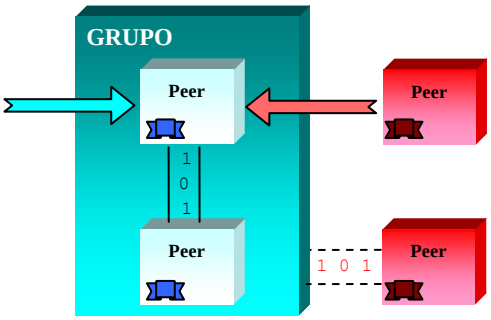

Figura 2. Seguridad orientada a grupos

La transparencia se consigue integrando la seguridad como parte del funcionamiento de los grupos dentro de SMEPP (ver Figura 2). Para entrar dentro de un grupo, es necesario presentar unas credenciales de seguridad. Si éstas son correctas, el nodo podrá participar en las comunicaciones seguras internas a ese grupo. El desarrollador de la aplicación necesita únicamente incluir esas credenciales al tratar de conectarse a un grupo, y el middleware SMEPP se encargará de implementar la admisión en el grupo y la comunicación segura dentro de ese grupo. Para entrar en un grupo, el middleware utiliza las credenciales de seguridad para ejecutar un mecanismo de autenticación mutua basado en protocolos de desafío-respuesta. La seguridad del canal de comunicaciones vendrá a su vez dada por una clave de sesión compartida por los miembros de un mismo grupo y utilizada por primitivas de criptografía de clave simétrica y funciones hash. Al mismo tiempo, existen otros mecanismos de protección transparentes al usuario que, por ejemplo, permiten renovar la clave de sesión cuando sea necesario, detectar si un miembro del grupo se está comportando de forma maliciosa (p. ej. eliminando mensajes del proceso de enrutado), etc.

	Nivel 0	**Nivel 1**	**Nivel 2**
Admisión en Grupo	Nada	Clave secreta compartida	Criptografía de clave pública
Seguridad en Datos	Nada	Autenticación	Autenticación y cifrado
Protección Clave de Sesión	Nada	Global (Refresco de clave)	Global y Local (anti-SCA)

Tabla 1. Niveles de seguridad dentro de SMEPP

Con respecto a la adaptabilidad, la seguridad no viene impuesta al desarrollador, sino que le es posible elegir el nivel de seguridad que desea y que se adapte mejor a los requerimientos de su aplicación (ver Tabla 1). SMEPP permite la configuración tanto del proceso de admisión en un grupo (utilizando claves compartidas o criptografía de clave pública) como de la seguridad en el envío de información (asegurar la integridad y la autenticidad de un mensaje o proteger la confidencialidad del mensaje) y la protección de las claves de sesión (refresco de claves o protección contra "*Side Channel Attacks*").

Además, SMEPP permite una granularidad extra a la hora de proteger las comunicaciones, los dominios de seguridad. Una red SMEPP dispone de múltiples dominios de seguridad: Un dominio global (comunicaciones entre los miembros de una red SMEPP) y varios dominios de grupo (comunicaciones entre los miembros de un mismo grupo). SMEPP permite así elegir para cada dominio el nivel de seguridad que se desee. Mediante la protección del dominio global, es posible elegir si una red SMEPP es accesible por cualquier dispositivo o sólo accesible por aquellos dispositivos autorizados a hacerlo. Por otro lado, es posible modificar el nivel de seguridad utilizado por un dominio de grupo en particular, permitiendo la existencia de grupos cuyos servicios sean de acceso público y de grupos con un grado de protección variable.

Finalmente, dado que la arquitectura de SMEPP sigue un modelo basado en componentes, es posible adaptar el middleware a la funcionalidad ofrecida. De esta forma, si no se utilizan determinados mecanismos de seguridad, no es necesario incluirlos. Esto permite un uso óptimo de los recursos disponibles. El uso de componentes también permite la reutilización de los mecanismos seguros: un mismo grupo de componentes facilita los servicios necesarios para implementar la seguridad de

los dominios de grupo y del dominio global. Respecto a las primitivas de seguridad a implementar dentro de esos componentes, las restricciones inherentes a los dispositivos de baja capacidad podrían suponer un problema. No obstante, los dispositivos de menor capacidad (nodos sensores) son capaces de ejecutar criptografía simétrica por hardware [7] o por software [8], y también son capaces de ejecutar criptografía de clave pública mediante curvas elípticas [9].

IV. APLICACIONES DE VALIDACIÓN

Para la validación de este middleware se han propuesto dos ámbitos de aplicación: uno en el campo del control radiológico en una instalación nuclear y otro en el de servicios para el hogar digital utilizando dispositivos móviles. Aunque parecen entornos muy diferentes, las funcionalidades que necesitan son muy similares en cuanto a control de dispositivos, generación de alarmas, videoconferencia, intercambio de mensajes entre usuarios… En ambos escenarios las alarmas se pueden generar de forma manual (lanzada por un usuario que detecta una anomalía), o bien de forma automática (generada a partir de los parámetros tomados a través de los sensores incorporados en los dispositivos). Se desencadena entonces una comunicación entre los distintos usuarios del sistema que puede ser a base de mensajes de texto, envío de imágenes o estableciendo una llamada de voz sobre IP o, incluso, una videoconferencia.

La aplicación de validación del campo nuclear [10] consta de dos partes, la primera dedicada al Control Dosimétrico Remoto de Trabajos (CRT) y la segunda orientada al Control Radiológico Medioambiental (CRM).

El CRT integra, en una única herramienta, datos que proceden tanto de equipos de dosimetría de área como de dosimetría personal, permitiendo disponer de toda la información de campo disponible, incluyendo vídeo y audio, así como otras posibilidades como el acceso a procedimientos, la identificación de personal con sistemas biométricos, etc. Todo ello proporciona una gran mejora tanto en la seguridad nuclear, a través de la mejor protección radiológica del trabajador, como en la eficiencia, gracias a un mayor control y supervisión del trabajo. Estas herramientas tienen aplicación en diversas circunstancias (operación normal, parada de recarga, emergencia (apoyo al Plan de Emergencia Interior, PEI), formación), en diversas áreas de trabajo (operación rutinaria en centrales nucleares, trabajos de descontaminación y desmantelamiento...), así como en otras instalaciones del ciclo de combustible nuclear.

El CRM está orientado para su uso tanto en operación normal, como su aplicación específica en caso de plantearse una emergencia, de forma que se define un escenario en el que se dispersan sensores de radiación con capacidades inalámbricas en el entorno de la central nuclear, permitiendo integrar medidas de radiación junto con otras variables de interés (meteorológicas básicamente), produciendo un resultado muy interesante tanto en su aplicación en la operación normal de la instalación como en situación de emergencia. Esta parte de la aplicación tiene además una traducción directa en otros campos como la protección medioambiental, de infraestructuras, frente a la amenaza terrorista o al tráfico ilícito de material radioactivo.

La integración de la aplicación CRT y la de CRM en una misma herramienta, produce como valor añadido la posibilidad de un seguimiento radiológico completo e integrado de la instalación y su entorno, en una única herramienta tecnológica. Así esta aplicación consiste en una red de sensores inalámbricos que da soporte a colectivos de trabajadores y personal de Protección Radiológica con el objetivo de reducir la radiación recibida por el personal y permitir la colaboración entre equipos de trabajo en entornos agresivos.

En primer lugar, se ha desarrollado un prototipo [11] con tres niveles de dispositivos de tratamiento de la información: 1) una red de motas micaZ de CrossBow basadas en ZigBee (802.15.4), 2) PDA con WiFi (802.11b/g) para su uso por personal

Figura 3. Dispositivos utilizados en el prototipo de monitorización ambiental

móvil de la instalación y 3) PC portátil con WiFi para personal de Protección Radiológica encargado de supervisar las condiciones ambientales y radiológicas de los equipos de trabajo. Dentro de este prototipo se ha conseguido la integración de las motas con sensores simples de actividad radiológica (contadores Geiger, ver Figura 3) y la medida de dicha actividad se presenta tanto en PDA como en PC. El próximo paso, en lo relativo al prototipo, es el diseño de un elemento sensor/transmisor integrado en un único componente y la inclusión de la captura y transmisión de audio y vídeo.

Dentro de las aplicaciones dedicadas a los servicios para el hogar digital, se está desarrollando, entre otras, una que consiste en que bien los usuarios o los propios dispositivos están capacitados para generar ciertos eventos, por ejemplo, una alarma. El sistema permite localizar al usuario que está en mejores condiciones de atender dicha alarma dependiendo del contexto y encaminar la alarma hacia ese usuario. Si la alarma no pudiera ser atendida por dicho usuario, el sistema se encarga automáticamente de encontrar un nuevo usuario que pueda atender la alarma. Una vez que se confirma la recepción de la alarma, los usuarios pueden intercambiarse mensajes de texto, imágenes o establecer una audio o videoconferencia.

En los desarrollos llevados a cabo hasta el momento, los usuarios son capaces de generar la alarma a través de una sencilla interfaz gráfica en un dispositivo móvil, por ejemplo, una PDA (ver Figura 4), y dicha alarma es recibida por el resto de usuarios de la red. Cuando un usuario confirma la recepción de la alarma, ambos usuarios (emisor y receptor de la alarma) pueden intercambiarse mensajes de texto, imágenes o audio. El resto de los usuarios son informados, a su vez, de que la alarma ya ha sido atendida.

Figura 4. Prototipo de aplicación de atención de alarmas y comunicación entre miembros de un grupo, por ejemplo, una familia.

En versiones posteriores de la aplicación, el aviso de alarma no llegará a todos los usuarios de la red, sino que el sistema será capaz de detectar en tiempo real y dependiendo del contexto, quién es el usuario más óptimo para atender dicha alarma. Si la alarma no puede ser atendida en un plazo de tiempo establecido (que puede depender del tipo de mensaje enviado), el aviso será encaminado hacia otro usuario y así sucesivamente hasta encontrar al usuario que pueda atender la demanda. De esta forma, no "se molesta" de forma indiscriminada a todos los usuarios del sistema, mejorando la percepción de la aplicación desde el punto de vista del usuario y optimizando las comunicaciones de la red (evitando *spam* de todos los mensajes), a la vez que se mejora también la aplicación de emergencia en sí, ya que el mensaje no es atendido por el primer usuario que decide aceptar la alarma, sino por el que el sistema cree más conveniente en cada momento. Además las alarmas podrán ser generadas de forma automática, por ejemplo, a través de un sensor de la vivienda que envía la incidencia a la pasarela residencial y ésta es capaz de difundirla por la red SMEPP. Los sensores que integrará la aplicación son de lo más heterogéneos, yendo desde la típica sonda que detecta una fuga de agua o gas, hasta dispositivos biométricos capaces de detectar la tensión arterial del usuario, generando una alarma si los valores se salen de un margen previamente establecido.

Utilizar el middleware de SMEPP en estas dos aplicaciones de alto nivel posibilita, por ejemplo, que dentro de una familia o de un grupo de trabajo sean los propios usuarios los que atienden las emergencias de los miembros de su grupo, con lo que se evita que detrás tenga que existir un centro supervisor que actúe como elemento centralizador de las peticiones de emergencia. Yendo a un caso práctico concreto como, por ejemplo, un servicio de teleasistencia, hasta ahora el modelo seguido era cliente/servidor, donde un usuario pulsaba un botón de emergencia y un centro de asistencia respondía a esa solicitud. Con este nuevo método de comunicación entre iguales, esa llamada de emergencia puede llegar al hijo de la persona que realiza la petición, pasando a ser un servicio mucho más personal, más social, más efectivo y, por qué no decirlo, de menor coste pues no tiene por qué haber una empresa dando el servicio de teleasistencia, sino que son los propios miembros de la familia los que reciben las peticiones. La comunicación entre iguales hace que la importancia del sistema radique en las propias personas, no en la infraestructura de la red, puesto que son los propios usuarios los que ofrecen los distintos servicios y los ponen a disposición del resto.

V. Conclusión

No hay pocos grupos de investigación trabajando en el desarrollo de un middleware genérico y reconfigurable para que los sistemas sean lo más eficientes y reutilizables posible. Ya no son tantos los grupos que trabajan en un middleware especialmente diseñado para dispositivos con baja capacidad, cuando cada vez son más los dispositivos "simples" (en el

sentido de tener pocas funcionalidades) pero con capacidades de comunicación con otros elementos que les rodean. Mucho menos son los que están pensando que esa comunicación puede ser a través de una red *ad-hoc* y bajo el paradigma *peer-to-peer*. El mayor reto ante todo esto es el de la seguridad. Una red que se establece de forma dinámica y sin ningún tipo de elemento centralizador no puede aplicar ninguna de las técnicas de seguridad tradicionales.

En este artículo se ha presentado el trabajo realizado en el proyecto SMEPP, donde se está desarrollando un middleware para sistemas seguros que tiene en cuenta estas características, permitiendo el desarrollo de sistemas P2P seguros sobre un amplio rango de dispositivos y plataformas. Actualmente se encuentran en funcionamiento los primeros prototipos del núcleo, así como los diferentes componentes de la arquitectura. El lenguaje de implementación utilizado es Java y los requisitos mínimos para la ejecución del middleware no son muy altos (en torno a unos pocos kilobytes de memoria), puesto que es necesario su funcionamiento en dispositivos con bajos recursos como pueden ser *smartphones* o PDA. Estas restricciones se están teniendo en cuenta en la propia implementación realizada, limitando, por ejemplo, el número de hebras creadas o memoria consumida en el núcleo. Para el caso de dispositivos con capacidades aún menores, como las motas, el lenguaje de implementación es nesC [12] sobre TinyOS [13], trabajando en una versión simplificada del middleware que pueda interactuar totalmente con el resto de dispositivos SMEPP.

En este proyecto, cofinanciado por el programa IST-FP6, participan 8 socios, entre los que se encuentran empresas como Siemens, VTT, Tecnatom y Telefónica Investigación y Desarrollo, además de varias universidades y centros de investigación. El líder del proyecto es la Universidad de Málaga cuyos departamentos involucrados cuentan con importante experiencia en el desarrollo de proyectos relacionados con protocolos P2P y la seguridad. Se está trabajando para el despliegue del middleware en dispositivos de baja capacidad (sensores y actuadores embebidos de bajo coste y reducido consumo), así como en dispositivos móviles (PDA y *smartphones*) y pasarelas. El proyecto tiene una duración de 3 años y acabará en septiembre de 2009.

AGRADECIMIENTOS

Este artículo forma parte del trabajo realizado en el proyecto IST SMEPP (IST-5-033563), financiado por la Unión Europea a través del 6º Programa Marco dentro de la línea de actuación de tecnologías para la sociedad de la información. El trabajo del autor Rodrigo Román Castro ha sido financiado por el Ministerio de Educación y Ciencia de España bajo el Programa Nacional de Formación de Profesorado Universitario.

Los autores quieren además agradecer a sus compañeros en el proyecto, sus comentarios y contribuciones en la discusión de estas ideas. Las opiniones expresadas en el artículo son las de sus autores, y no representan necesariamente las de todo el consorcio del proyecto SMEPP.

REFERENCIAS

[1] http://www.smepp.org

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. *"Wireless sensor networks: a survey"*. Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 38, no. 4, pp. 393-422, March 2002.

[3] Costa, P. et al. The RUNES Middleware: A Reconfigurable Component-based Approach to Networked Embedded Systems. 16th Annual IEEE Internacional Symposium on Personal Indoor and Mobile Radio Communications IMRC'05), Berlin, Germany. September 2005.

[4] The EU FP6 MORE project, http://www.ist-more.org/

[5] Georgantas, Nikolas; Ben Mokhtar, Sonia; Bromberg, Yérom-David; Issarny, Valérie; Kalaoja, Jarmo; Kantarovitch, Julia; Gérodolle, Anne; Mevissen, Ron. 2005. The amigo service architecture for the open networked home environment. Proceedings of 5th Working IEEE/IFIP Conference on Software Architecture. WICSA. Pittsburgh, 6 - 10 Nov. 2005 (http://www.hitechprojects.com/euprojects/amigo/deliverable.htm)

[6] http://www.osgi.org

[7] IEEE 802.15 WPAN™ Task Group 4 (TG4). http://www.ieee802.org/15/pub/TG4.html

[8] K. Jun Choi, and J.-I. Song. *"Investigation of Feasible Cryptographic Algorithms for Wireless Sensor Network"*. Proceedings of the 8th International Conference on Advanced Communication Technology (ICACT 2006). Phoenix Park (Korea), February 2006.

[9] An Liu, and Peng Ning, *"TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks"*. Technical Report TR-2007-36, North Carolina State University, Department of Computer Science, November 2007.

[10] E. Cabrera, R.J. Caro, M. Díaz, J. Serrano "*Proyecto SMEPP: Redes inalámbricas de sensores y sistemas "peer-to-peer" empotrados. Aplicación en la industria nuclear*" Ponencia de la 33 Reunión anual de la Sociedad Nuclear Española (SNE 2007)

[11] R.J. Caro "*SMEPP un proyecto I+D+i del 6º PM aplicado al Control Radiológico Medioambiental y a la mejora de la Protección Radiológica"* (reseña) Radioprotección Revista de la Sociedad Española de Protección Radiológica. Nº 54, vol. XIV, 2007.

[12] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, D. Culler, "The nesC language: a holistic approach to networked embedded systems", in: Proceedings of the ACM SIGPLAN conference on programming language design and implementation (PLDI 2003), San Diego, CA, USA, June 2003, pp. 1–11.

[13] TinyOs Community Forum, http://www.tinyos.net