Experiencias en entornos de Computación Ubicua mediante Arquitecturas Orientadas a Servicios

Juan Antonio Álvarez, Manuel David Cruz, Alejandro Fernández, Juan Antonio Ortega, Jesús Torres.

Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla.
Avda. Reina Mercedes s/n. Sevilla (Spain)
juan@lsi.us.es, {manuel3086,alejandro.fdez}@gmail.com, {ortega,jtorres}@lsi.us.es;

Resumen

El desarrollo del proyecto Domoweb, cuyo objetivo principal, es la creación de un entorno domótico gestionado vía Web, nos ha permitido acumular experiencia sobre las arquitecturas orientadas a servicio, concretamente sobre la plataforma basada en la especificación de OSGi e implementada con software de fuentes abiertas. En este artículo, comentamos las decisiones tomadas durante el desarrollo del proyecto y ampliamos la visión inicial hacia ámbitos de computación ubicua más genéricos que el hogar digital. Además, señalamos la necesidad de integrar en la arquitectura propuesta, la gestión de los Servicios Web, ampliando de ese modo las capacidades del entorno y su funcionalidad.

1. Introducción

Hoy en día, en la sociedad de la información, la expansión de las nuevas tecnologías en todos los ámbitos, ha generado un aumento de la demanda de servicios de gestión y control de nuestro entorno. Además, dichos servicios, tienen que estar disponibles desde cualquier punto con conexión a Internet. Debido a ello, surgió el proyecto Domoweb: "Metodologías para el diseño y desarrollo de sistemas domóticos controlados vía Web" cuya finalidad es la construcción de un sistema domótico que ofrezca soporte a amplia gama de tecnologías domóticas, cuyo control es realizado vía Web con independencia del dispositivo desde el que se acceda.

Aunque existían otras propuestas [3,13], en la implementación de dicho proyecto se ha decidido usar una arquitectura orientada a servicios (SOA) basada en la especificación de OSGi [14], en la que se propone una arquitectura abierta y común para los proveedores de servicios, desarrolladores, operadores de telecomunicaciones y fabricantes de dispositivos para desarrollar, desplegar y gestionar servicios de una manera coordinada. En la versión 3 de dicha especificación [15], se detallan algunos modelos de referencia y los servicios que son necesarios para la infraestructura de la pasarela.

En [4] se explica como interpretar la arquitectura orientada a servicios. A modo de guía mostramos dicha estructura en la figura 1.

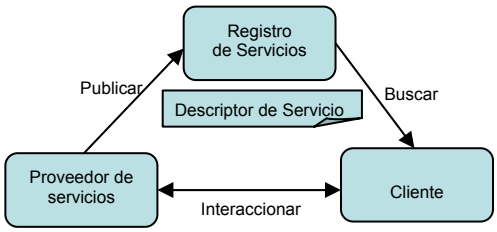

Figura 1. Patrón de interacción orientado a servicio.

A nivel internacional, Prosyst [16] lidera el mercado, mientras que a nivel nacional Telefónica I+D [17] y Telvent [18] se encuentran en una posición preferente para la comercialización y explotación de productos para el hogar digital. Debido a que usan software propietario, resultan soluciones caras. En el artículo comentamos como usar el software de fuentes abiertas para dar una solución eficiente y económica.

En el siguiente apartado, comentaremos las experiencias obtenidas durante las etapas de diseño y desarrollo del proyecto Domoweb bajo una arquitectura orientada a servicios. En el apartado de conclusiones y trabajo futuro explicamos algunas carencias que hemos encontrado y propuestas para la mejora de la plataforma, así como posibles ámbitos de aplicación de la misma basados en la computación ubicua. Resumiremos nuestras experiencias en el desarrollo y propondremos vías de nuevas investigaciones, entre las que se encuentra la adaptación a Servicios Web.

2. Trabajos realizados

El objetivo primordial de Domoweb, es el de diseñar y desarrollar un sistema domótico virtual, controlado vía Web y basado en fuentes abiertas. Con este nombre, definimos un sistema capaz de gestionar elementos típicos de la domótica como pueden ser sistemas de alarmas y sistemas de control del confort del usuario, mediante los sensores y actuadores necesarios para ello. Una de las particularidades de este sistema es su capacidad de ser gestionado virtualmente, a través de interfaces adaptativas por el usuario final de manera fiable, segura y fácil. De este modo además de poder realizar operaciones de control y supervisión desde dentro de la vivienda, se podrían realizar desde cualquier punto con acceso a Internet. Dichas operaciones suponen interaccionar con dispositivos de diversos protocolos de comunicación como X-10 [20] ó tecnologías como USB [10] o Bluetooth [12]. A continuación veremos una a una las decisiones de diseño e implementación más importantes, como el modelo de negocio elegido, la implementación de OSGi empleada y los servicios implementados.

2.1. Modelo de negocio

Una vez estudiado el dominio del problema y teniendo en cuenta los objetivos del proyecto, se analizaron los modelos de referencia propuestos en OSGi R3. Finalmente, se hizo una adaptación del modelo autogestionado, que según la representación que establece OSGi, quedó como se muestra en la figura 2.

En nuestro modelo el *Service User* asume los roles de operador, propietario y proveedor de servicios (como en el modelo autogestionado), esto quiere decir que controla el sistema y será el encargado de decidir qué servicios se proporcionan a los clientes. Partiendo de esta definición debemos diferenciar entre el *Service User* y un usuario final o *End User* ya que este último será quien use estos servicios, ya sea localmente haciendo uso del PC (SPS) directamente, o bien accediendo a la pasarela de servicios remotamente a través de un proveedor de red. Además el *Service User* puede adoptar el rol de *End User* para hacer uso del sistema, probar nuevos servicios, etc.

El *Service Platform Server* (SPS) incluirá al *Service Deployment Manager*. En nuestro caso el *Service Platform* será la implementación de OSGi y el *Service Deployment Manager* su GUI. Los dos servicios ejecutándose sobre un PC (SPS) que hará de servidor convencional.

Por otro lado, no es necesario que el *Operator* se encuentre en la misma red local que el *Service Platform Server*.

El SPS representará una pasarela de comunicaciones. A través del SPS los dispositivos locales conectados a una red local podrán comunicarse con una red externa (a través de un *Network Provider*). En este caso el SPS asumirá el rol de pasarela de servicios (*Service Gateway*), esto quiere decir que un usuario remoto podrá usar los servicios que proporcione la pasarela de servicios.

Cuando hablamos de Owner nos referimos al propietario del SPS. Por Service Provider entendemos aquella entidad que provee de servicios a la pasarela, es decir, que desarrolla y despliega estos servicios.

2.2. Elección de implementación

Debido a que nuestro proyecto debe basarse en software de fuentes abiertas y que la especificación de OSGi es demasiado extensa para implementarla con los recursos humanos disponibles, optamos por utilizar implementaciones existentes de software libre. En concreto nos han sido de mucha utilidad OSCAR [5] y Knopflerfish [19], que siguen la especificación de OSGi R3.

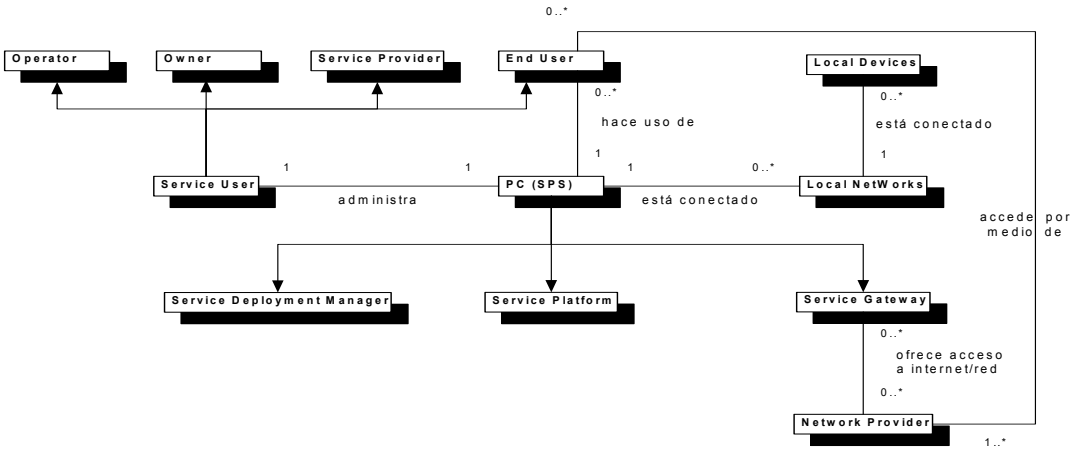

Figura 2. Modelo de negocio escogido.

OSCAR es muy estable, y es usada por multitud de desarrolladores y programadores en distintos proyectos. El framework está pensado para experimentación de software orientado a servicios en general. OSCAR puede ser fácilmente incluido en otros proyectos y usado como un mecanismo de extensión. Gracias a OSCAR, podemos realizar ensamblado dinámico de aplicaciones, añadiendo y extendiendo funcionalidades de una aplicación base en tiempo de ejecución, descargando sobre él nuevos componentes y servicios. Además proporciona técnicas de aseguramiento de la calidad de las aplicaciones que funcionan bajo ese entorno dinámico y basado en componentes.

Knopflerfish es otro desarrollo en código libre del framework especificado en OSGi R3. Como en el caso de OSCAR, Knopflerfish está orientado a componentes y facilita la experimentación de software orientado a componentes y servicios.

Ambos proyectos presentan una interfaz muy intuitiva desde la que se nos permite instalar, desinstalar, iniciar, parar y actualizar componentes, además proveen repositorios para componentes que pueden analizarse ya que se facilita el código fuente.

2.3. Implementación de servicios

OSCAR y Knopflerfish ofrecen diferentes servicios, tanto de infraestructura como de aplicación, que nos son útiles. Destacamos los siguientes:

- *Log Service*: este servicio provee la interfaz necesaria para que se puedan registrar mensajes en la pasarela por parte de los servicios que lo requieran.
- *User Admin Service*: como base para la implementación de un servicio de gestión de usuarios.
- *Service Tracker*: con este servicio se puede llevar a cabo un proceso de vigilancia de servicios. Esto es muy útil para prevenir fallos por problemas de dependencias entre componentes.
- *Configuration Admin*: este servicio es el encargado de configurar los bundles (componentes OSGi) que requieran una

configuración cuando son desplegados. De esta manera un *bundle* puede configurar a otro a través de este servicio.

- *HTTP Service*: ofrece un servidor web en el que se pueden registrar tanto *servlets* como recursos de manera que sean accesibles desde el exterior.

Además de estos servicios, fue necesario implementar otros que pasamos a describir.

2.4. Servicios de infraestructura

Para el desarrollo del sistema fue necesario elaborar un servicio de comunicación con sistemas de bases de datos. Dicho servicio, gestionaría el almacenamiento de manera persistente de la información relacionada con dispositivos, usuarios y el espacio domotizado. De entre las soluciones de almacenamiento persistente, se eligió el uso de un Sistema de Gestor de Bases de Datos Relacional (SGBDR) debido a que es una tecnología muy extendida en entornos web y en arquitecturas cliente servidor.

Las soluciones Java para la comunicación con SGBDR's pasan por el uso de la API *Java Data Base Connectivity (JDBC)* [9]. *JDBC* provee conectividad con bases de datos basadas en SQL con independencia de la plataforma concreta que usemos (*Oracle*, *MySQL*, etc.). De esta manera tratamos de crear un enlace entre los servicios desde el punto de vista de *OSGi* y el servicio que provee *JDBC*.

Aunque la comunicación con bases de datos a través de *JDBC* es muy sencilla presenta inconvenientes como el uso de *drivers* específicos para cada plataforma, por lo que es necesario que los distintos fabricantes los faciliten.

Debido a que Domoweb trata siempre de hacer uso de software de fuentes abiertas este servicio incluye el *driver* necesario para la comunicación con *MySQL* y por tanto será éste el SGBDR empleado por los distintos servicios implementados.

Para los usuarios experimentados y acostumbrados a usar este tipo de tecnologías, se ha implementado un servicio a través del cual la pasarela almacena un detallado registro de todo lo que sucede en la misma. Gracias a esto el usuario será capaz de saber en todo momento qué procesos se han llevado a cabo en su pasarela. Por ejemplo, se puede saber cuales han sido los últimos usuarios que han accedido al sistema, los dispositivos que se han puesto en marcha a una determinada hora, los intentos de intrusión a través de la red, etc.

Como servicio de infraestructura básico se ha incluido un sistema de gestión de usuarios. Con él somos capaces de controlar todo lo referente a altas y bajas, y también todo lo concerniente a clasificación de los mismos y su autentificación en el sistema. Además, cada vez que un usuario efectúa una petición debe comprobarse si está autorizado a ello, dependiendo de la categoría a la que pertenezca. Esto último también es controlado por el servicio de gestión de usuarios. OSGi propone, en su tercera especificación, un servicio de administración de usuarios, *User Admin Service*, en el que se detalla un modelo de autorización basado en rol (llamemos rol a una categoría de usuarios, por ejemplo, administrador, usuario, niños, adultos, etc., o también a un usuario en concreto) que nosotros hemos empleado por adaptarse perfectamente a nuestros requerimientos. A grandes rasgos, el modelo sugiere que todo servicio que ofrezca la pasarela esté asociado a un grupo de usuarios (categoría) y el usuario que quiere hacer uso de dicho servicio deberá pertenecer (o incluso ser él mismo, si el rol fuese una persona) a éste.

La figura 3 muestra un ejemplo de gestión de grupos, usuarios y las autorizaciones por grupos en un hipotético hogar digital.

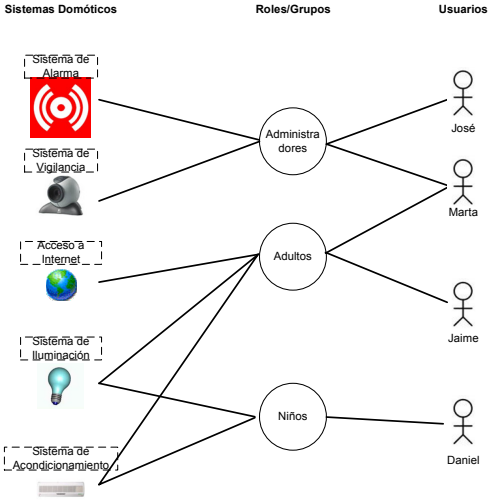

Figura 3. Gestión de usuarios

En cuanto al método de autentificación, Domoweb ha optado por el mecanismo convencional de *login* y *password,* aunque la implementación queda abierta a cualquier otro sistema debido a la gran variedad de éstos que hay actualmente en el mercado.

También podemos considerar como servicios de infraestructura aquellos que nos permiten la gestión y administración de dispositivos, sea cual sea la tecnología de estos. Domoweb ha implementado servicios que se comunican con dispositivos de tecnologías *X10*, *USB* o *Bluetooth*. La pasarela aparece en este caso como medio de interconexión de redes de distintas tecnologías. OSGi proporciona un modelo de acceso a dispositivos. La idea fundamental de este modelo es facilitar el acceso a los dispositivos a través de un conjunto de *drivers* que se encuentran jerarquizados desde un nivel de abstracción muy bajo a un nivel muy alto. Esto anima, por un lado, a los distintos fabricantes a desarrollar *drivers* de bajo nivel compatibles con el modelo de OSGi y, por otro, a terceras partes a implementar *drivers* de un nivel de abstracción superior con el objeto de facilitar la integración de todas las tecnologías. En la figura 4 podemos observar la especialización de *drivers*.

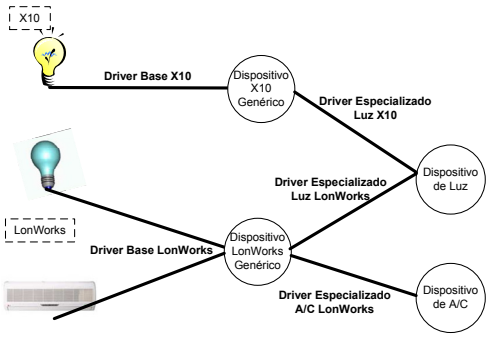

Figura 4. Especialización de drivers.

Debido al dinamismo que el entorno de ejecución de OSGi propone, consideramos muy importante determinar la manera y el orden en el que los distintos servicios deben desplegarse en la pasarela durante el inicio del sistema.

Al proceso de descarga e instalación de los elementos necesarios para que nuestro sistema pueda comenzar a funcionar correctamente, lo denominaremos *aprovisionamiento inicial*.

Con tal fin, en Domoweb, hemos empleado el *Oscar Bundle Repository* (OBR)[6]. OBR es un servicio que permite la descarga e instalación de servicios de manera remota a través de un repositorio descrito en XML. Los distintos repositorios pueden ser accedidos de dos formas, a través de la web o mediante un acceso programado en la pasarela haciendo uso de algún servicio que lo haga posible. Es responsabilidad del operador de la pasarela decidir qué repositorio de servicios será el empleado en cada caso e indicar si se tiene autorización para poder acceder a ellos y poder usarlos.

La necesidad de garantizar que el sistema arranca de manera adecuada cuado se inicia por primera vez, nos obliga a implementar un servicio de puesta en marcha (*StartupService*). Éste, descarga los componentes software básicos o de infraestructura que necesita el sistema para comenzar a funcionar. Para ello realiza las peticiones pertinentes a *OBR* y es éste último el que realiza la descarga e instalación de los componentes solicitados.

En el caso de nuestro estudio, debido a las funcionalidades que caracterizan cada servicio, se decidió desplegar los componentes en un determinado orden. En primer lugar se despliega el servicio de comunicación con las bases de datos. A continuación el servicio de registro de eventos en el sistema. Y por último los servicios de administración de usuarios y de control y gestión de dispositivos.

Estos servicios básicos o de infraestructura son a su vez dependientes de otros servicios básicos de OSGi, lo que obliga a *OBR* a resolver estas dependencias.

StartupService ha sido también añadido al repositorio aunque no sea considerado un servicio de utilidad final para el usuario.

2.5. Servicios de aplicación

Como ya comentamos con anterioridad, dentro de este conjunto de servicios podemos encuadrar todos aquellos que ofrezcan una funcionalidad a los usuarios finales del sistema. Quizás uno de los servicios más útiles sea el de gestión de los dispositivos conectados a la pasarela. Éste, permite a los usuarios controlar y supervisar el estado de los mismos de una manera homogénea e independiente de la tecnología gracias al modelo

de acceso a dispositivo de OSGi. También tiene en cuenta qué usuarios pueden hacer uso (tienen autorización) de qué dispositivos. La información sobre las autorizaciones se recoge a través del servicio de infraestructura para la administración de usuarios comentado en el apartado anterior.

Muy relacionado con la gestión de los dispositivos se encuentra el servicio de gestión de escenas. Una escena es una combinación o sucesión de eventos (sucesos) que se desarrollan en un espacio domotizado, por tanto determina el estado en el que quedan los dispositivos del sistema. Escenas típicas pueden ser por ejemplo: "llegada a casa" (cuando algún residente entra a la vivienda, se encienden las luces necesarias para la visibilidad, se suben las persianas, se reproducen los mensajes del contestador, etc.); "visualización de películas" (se apagarían las luces y se encendería la TV. etc.), "marcharse de vacaciones" (se activa la alarma y la simulación de presencia) etc. Gracias a este servicio, los usuarios pueden añadir configuraciones de dispositivos personalizadas para cubrir sus necesidades.

Por otro lado, contamos con un servicio de creación y configuración del espacio en el que el usuario podrá indicar al sistema las dimensiones de su vivienda así como la localización de los dispositivos, asignando nombres comunes a éstos. La configuración del espacio se detallará más adelante en la descripción del lenguaje de especificación de casas.

2.6. Interfaces de usuario

Todos los servicios que provee la pasarela, podrán ser accedidos a través de interfaces web tanto desde el interior como desde el exterior de la casa. Para ello se ha creado un sitio web basado en tecnología de *servlets* java. Este conjunto de *servlets* son albergados por la pasarela residencial y funcionan bajo el servicio *http* propuesto en OSGi.

De esta manera, hemos unificado y homogeneizado la manera en la que los usuarios controlan su vivienda, ya que, mediante peticiones HTTP se pueden instalar y lanzar todos los servicios.

Desde el interior de la vivienda, la interfaz podría mostrarse en dispositivos móviles con alguna tecnología *wireless* o incluso a través de la pantalla de televisión convencional.

Desde el exterior la conexión se establece a través de Internet por lo que es fundamental que la vivienda domotizada cuente con una conexión. La pasarela podrá mostrar páginas web a todo dispositivo que cuente con un navegador web.

En la figura 5 podemos apreciar la interfaz que se le presenta a un usuario cuando desea gestionar las escenas:

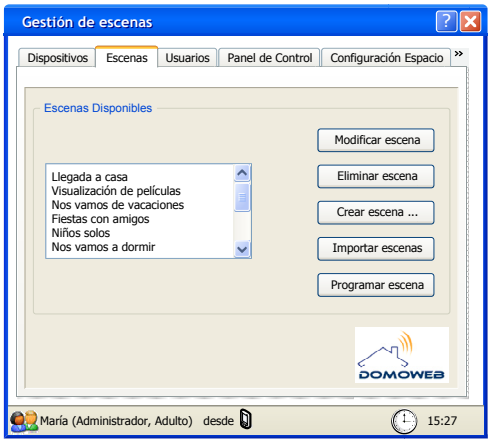

Figura 5. Interfaz para gestión de escenas

2.7. Lenguaje de descripción de casa.

Para describir la configuración del espacio (dimensiones, distribución y lugar que ocupan los dispositivos) hemos creído conveniente la creación de un pequeño lenguaje XML al que hemos llamado *lenguaje de descripción de casa*.

El lenguaje queda especificado por un *DTD* (o por un *XMLSchema)*. En él, se describe la vivienda a partir de distintos niveles jerárquicos:

1. Planta: cada una de las plantas de la vivienda. Queda definida por un identificador, un nombre común y un conjunto de puntos que forman los límites de la misma (un mínimo de 3).
2. Estancia: la planta se divide en estancias. Cada una tiene asociado un identificador, nombre común, puntos que la delimitan y los dispositivos que se encuentran en ella. Además podemos asignar una categoría a cada una (dormitorio, salón, jardín etc.).

3. Dispositivo: los dispositivos se encuentran descritos por un identificador, un nombre común y la posición que ocupan dada por dos puntos. Además podemos asignar un tipo a cada uno de ellos para tener conocimiento de qué actuaciones podemos realizar sobre los mismos.

A partir de este lenguaje podemos automatizar tareas de creación de imágenes de las viviendas. Integrando estas imágenes con una interfaz web se consigue que el usuario vea representada de una manera gráfica los espacios de su vivienda y por tanto una interacción mas intuitiva. La figura 6 muestra un ejemplo de interfaz en la que está representada una vivienda ficticia.

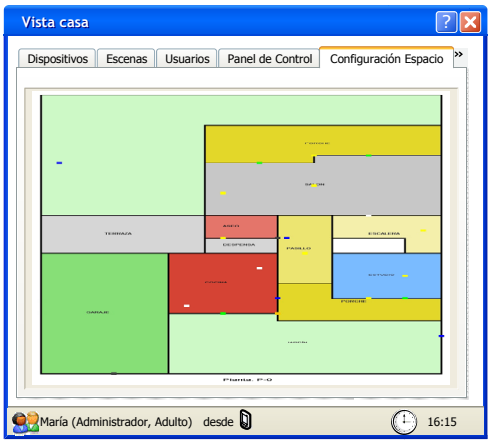

Figura 6. Interfaz de manejo de vivienda.

3. Conclusiones y trabajo futuro

Observando la domótica desde el punto de vista de la Inteligencia Ambiental, es decir, no mirándolo sólo desde el punto de vista de la casa inteligente, un tema de interés a tratar dentro de este mundo sería el campo del aprendizaje automático, campo que no se centra sólo en el mero desarrollo tecnológico. Han resultado muchos estudios sobre este tema tratándose en ellos de diseñar una arquitectura de aprendizaje que se adapte a las necesidades de la gente originadas a partir de las vivencias obtenidas día a día dentro de un entorno digital. Esto no es algo sencillo ya que los gustos y costumbres de las personas pueden diferir mucho unas de otras y además tienen la dificultad añadida de que el aprendizaje debe hacerse de una forma no intrusiva [11].

El sistema propuesto por Domoweb se basa en la interacción con los usuarios. Responde a estímulos, llegados a través de sensores, producidos por actuaciones de éstos dentro del espacio domotizado. Por lo tanto la interacción con el usuario es esencial. Mirándolo desde este punto de vista, una de las posibles líneas de investigación dentro del razonamiento cualitativo, que se podría abarcar desde Domoweb, es la de crear una arquitectura capaz de llevar a cabo el aprendizaje de forma autónoma, de acuerdo a las necesidades del usuario; y no sólo el aprendizaje sino también la respuesta a las necesidades de los mismos de manera automática. Tal arquitectura debe adaptarse a los deseos de los usuarios mediante la observación de sus acciones, siempre de una manera no intrusiva.

En segundo lugar, la falta de interacción con Servicios Web de nuestro proyecto, supone un problema a la hora de contar con servicios menos orientados al hogar digital y más enfocados a entornos abiertos de computación ubicua. Sería interesante conseguir procesar los mensajes de tipo *SOAP* para poder obtener funcionalidad añadida. Ante esta situación, existe una serie de servicios en el repositorio de bundles de *Knopflerfish* (*SOAP desktop*, *SOAP FW* y *SOAP objects*) que permiten analizar mensajes *SOAP*, así como, *Apache Axis* [1], un componente, que permite exportar los servicios *OSGi* como Servicios Web, además de importar Servicios Web en un *framework OSGi*, aunque aún está en una versión primitiva (0.1.0).

Una vez conseguido el objetivo de mejorar la funcionalidad gestionando Servicios Web, habría que contemplar entornos mucho más amplios que el inicial (hogar digital o vivienda domótica). En nuestro proyecto, tras preparar un laboratorio en el Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Sevilla, en el que existe conexión *Wi-Fi*, estamos estudiando, como adaptar la tecnología descrita a entornos como sillas de ruedas preparadas para comunicaciones inalámbricas o incluso a computación *vestible* (*wereable computing*). No sería complicado si conseguimos integrar una importante variedad de sensores que podrían

comunicarse con la pasarela integrada en un dispositivo tipo PDA [21].

Por otro lado, aspectos como el método de autentificación pueden ser mejorados en el sistema propuesto por Domoweb. Por ello hemos contemplado la posibilidad de añadir otros mecanismos de autentificación como los basados en métodos de localización [2], además del que ya se está llevando a cabo, comentado en apartados anteriores, autentificación por identificación del usuario. El primero de ellos vislumbra la posibilidad de que el usuario que se encuentre dentro de un espacio preparado para computación ubicua pueda hacer uso de servicios de aplicación específicos que se encuentren en ese espacio sin necesidad de pasar por un proceso de identificación. Algunos ejemplos de estos servicios pueden ser los de televisión o telefonía. El estudio se centraría en los distintos mecanismos que existen en la actualidad para este fin, como por ejemplo, los sensores de movimiento, las cámaras de vigilancia o incluso la posibilidad de integrar un sistema de localización por GPS (en espacios al aire libre) o basados en *bluetooth*. La expansión, cada vez mayor, de dispositivos móviles de última generación con tecnología *bluetooth* incorporada podría suponer un despegue de este método de autentificación. De esta forma, aquella persona portadora de un dispositivo móvil que se adentrara en un espacio capaz de reconocer dispositivos que usen esa tecnología, sería automáticamente identificada.

En lo referente a las interfaces de usuario, se nos abre una importante línea de investigación en la que podamos ofrecer interfaces que se adapten al tipo de dispositivo que desee interactuar con la vivienda. Para ello podemos basarnos en el estándar CC/PP propuesto por W3C.

4. Referencias

[1] Apache Axis. http://ws.apache.org/axis/

[2] Bardram Jackob E., Kjaer Rasmus E, Pedersen Michael. Context-aware user authentication – supporting proximity-based login in pervasive computing. 2003.

[3] Doctor, F., Hagras,H.A.K., Callaghan,V. An Intelligent Fuzzy Agent Approach to Realising Ambient Intelligence in Intelligent Inhabited Environments. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans. 2004.

[4] Hall, Richard and Cervantes H.. "Challenges in Building Service-Oriented Applications for OSGi," IEEE Communications, Volume 42, Number 5, May 2004.

[5] Hall, Richard. Oscar, An OSGi framework implementation, http://oscar.objectweb.org

[6] Hall, Richard. Oscar Bundle Repository, http://oscar-osgi.sourceforge.net

[7] Highttower Jeffrey, Borriello Gaetano. Location systems for ubiquitous computing. 2001.

[8] Horowitz, Bruce, Magnusson Nils, Klack Niclas. Telia's Service Delivery Solution for the home. 2002.

[9] JDBC API. http://java.sun.com/products/jdbc/

[10] jUSB. Java API for USB. http://jusb.sourceforge.net/ 2003

[11] Lopez, A. Doctor, F. Sánchez, L. Algoritmo GA-P difuso para la generación de controladores en edificios inteligentes. 2004.

[12] Motorola. Wireless Software, Applications & Services. Java APIs for Bluetooth Wireless Technology (JSR-82). 2002.

[13] Moya, F., López J.C. SENDA: An alternative to OSGi for large scale domotics. 2002

[14] Open Source Gateway Initiative. OSGi. 1999.

[15] OSGi Alliance. OSGi Service Platform Release 3. 2003

[16] Prosyst. OSGi Device and back end software platform. http://www.prosyst.com/products/osgi.html

[17] Telefonica I+D. Hogar.es.

[18] Telvent y otros. OSMOSE

[19] The Knoplerfish Project, http://knopflerfish.org/index.html

[20] X10 – Interface Communication Protocol. http://www.maison-domotique.com/telechargements/x10/protocol_X10.pdf . 2003.

[21] Yuan, Michael J. "Enterprise J2ME", Prentice Hall, 2003.