Mashups en Ambientes Inteligentes

Diego A. Godoy[(1,a)];, Eduardo O. Sosa[(1,b)];

[(1)]Centro de Investigación en Tecnologías de la Información y Comunicaciones (C.I.T.I.C.). Universidad Gastón Dachary (U.G.D.)

Campus Urbano, UGD. Posadas, Argentina
Teléfono: +54-376-4438677

[(a)]diegodoy, [(b)]eduardo.sosa@citic.dachary.edu.ar

Resumen

La aparición de los mashups conjuntamente con la Inteligencia Ambiental abre nuevas posibilidades para el desarrollo de aplicaciones que se adapten al contexto de sus usuarios. Esto da lugar a la necesidad de nuevas arquitecturas en las que se posibiliten la creación de mashups que sean sensibles al contexto y que puedan ser creados de forma práctica por el usuario. Es por ello que en este trabajo se analizarán dos arquitecturas existentes para la creación de esta nueva clase de mashups y se discutirán posibles mejoras. El trabajo presentado aquí se realiza en el marco del proyecto de investigación denominado “Diseño de arquitecturas de soporte a la Internet del Futuro y Ambientes Inteligentes” que se ha formalizado por la resolución (19/A/12) de la Universidad Gastón Dachary.

Palabras clave: Mashups, Sensibilidad al Contexto, Inteligencia Ambiental

Contexto

El trabajo presentado aquí se realiza en el marco del proyecto de investigación denominado “Diseño de arquitecturas de soporte a la Internet del Futuro y Ambientes Inteligentes” que se ha formalizado por la resolución (19/A/12) del la U.G.D. mediante el llamado al 5to Concurso de Proyectos de Investigación en 2012. EL objetivo de este proyecto es "Diseñar arquitecturas de soporte a Internet del futuro y Ambientes Inteligentes para su aplicación a Ciudades Inteligentes". Como antecedentes y resultados parciales del mencionado proyecto se pueden indicar 3 proyectos de trabajos de grado en curso, correspondientes a la Carrera de Ingeniería en Informática de la U.G.D. y un trabajo final especialización Ingeniería de Software de la Universidad Nacional de la Plata. Indicar el proyecto en que está inserta la línea de I/D presentada y la/s instituciones que coordinan el proyecto. En los casos que corresponda indicar la Institución que acredita el proyecto y los organismos/empresas que contribuyen a su financiamiento.

Introducción

En los últimos años hemos sido testigos del gran desarrollo de dos tecnologías: los dispositivos móviles (basados en redes inalámbricas, con GPS y sensores) y la Web.

En cuanto a la primera de estas tecnologías se han desarrollado grandes avances en redes inalámbricas, como ser: protocolos más eficientes, mayores anchos de banda, mayores áreas de cobertura, nuevos tipos de sensores, etc. Además los dispositivos móviles cuentan cada vez más, con mayor poder de cómputo, pantallas más grandes y se consiguen a un menor precio. Estos avances facilitan el desarrollo de aplicaciones en donde la movilidad y ubicuidad son parte esencial.

Sin duda una de las características más importantes de los sistemas móviles, es que brindan la posibilidad de aprovechar la posición geográfica para asistir al usuario mientras este se desplaza. Ejemplos de ello son los sistemas que guían a los usuarios en las visitas a museos, sistemas de navegación basados en GPS, o proyectos como Friend Finder de A&T que notifica la presencia de personas geográficamente cercanas.

Hoy en día el desarrollo de sistemas para escenarios móviles ha aumentado considerablemente. Se ha pasado de aplicaciones casi experimentales o utilizadas solo en ambientes académicos, a algunas aplicaciones comerciales como las orientadas a negocios, el uso personal o el ocio.

De la misma manera que las redes inalámbricas, el aumento del poder de cómputo de los dispositivos y capacidad de censado de los dispositivos móviles supone una revolución, el crecimiento de la web también fue revolucionario en cuanto a la forma de presentar la información a los usuarios de sistemas informáticos. Esto dio origen a una evolución de la web, llamada web 2.0, en donde ya los usuarios no son únicamente consumidores, y donde la experiencia del usuario a usar estos sistemas se hizo más agradable, posibilitando personalizar las aplicaciones a las necesidades concretas de los usuarios y mejorar la interfaz gráfica.

Con estas mejoras de la web, muchas veces ya no son suficiente sitios donde se puedan obtener datos de una sola fuente, es más, se hace sumamente necesario contar con datos de diversas fuentes integradas, y para ello se han desarrollado plataformas que permiten hacer mezclas o “mashups” tanto de información como de comportamiento de las aplicaciones. La particularidad de estas mezclas es que son los mismos usuarios los que tienen la posibilidad de crearlas para cumplir con sus requisitos y no necesitan de un programador o conocer lenguajes como Java o C++.

La Inteligencia Ambiental (AmI) o Computación Ubicua no está ajena a los avances de estas dos tecnologías y se ve beneficiada por ambas. Estos avances permiten que la AmI se acerque cada vez más al cumplimento a la visión del Information Society Technologies Program Advisory Group (ISTAG) de personas rodeadas de interfaces inteligentes embebidas en objetos de la vida cotidiana que permitan la interacción de forma natural y sin esfuerzo con diferentes sistemas de información.

Este trabajo tiene como objetivo conocer cuáles son las plataformas existentes para la creación de mashups sensibles al contexto en entornos de AmI y proponer mejoras a estas.

La evolución y creciente miniaturización de los dispositivos de cómputo, que hace posible que pequeños procesadores y diminutos sensores se integren cada vez más en objetos cotidianos. Mark Weiser ya había previsto ésta evolución hace más de 10 años y la publicó en su artículo "The Computer for the 21st Century" [1]. En un trabajo posterior Weiser se planteaba los siguientes interrogantes “¿Cuál es la idea de la computadora del futuro? ¿El agente inteligente? , ¿La televisión (multimedia)?, ¿El mundo gráfico

3-D (realidad virtual)? ¿El equipo de voz ubicua de la película StarTrek?, ¿Las interfaces gráficas de usuarios –GUI-, elaboradas y seleccionadas?, ¿La máquina que por arte de magia cumpla y acceda a nuestros deseos?" [2]. La respuesta que él mismo ha dado es: ninguna de las anteriores. La sencilla razón es que todos estos conceptos comparten un defecto básico: se llevan a la práctica con algo tangible, visible.

Es así Weiser que definió el concepto de "computación ubicua", refiriéndose a las computadoras omnipresentes al servicio de las personas en su vida cotidiana, en el hogar y en el trabajo, y funcionando de manera invisible y no intrusita. Los principios enunciados originalmente por Weiser han sido validados por diferentes investigadores e ingenieros, convirtiéndose en un escenario real en ciertos ámbitos y extendiéndose a escala global [3].

En 1999, el Information Society Technologies Program Advisory Group (ISTAG) de la Unión Europea ha utilizado el término "inteligencia ambiental" del inglés Ambient Intelligence (AmI) de manera similar para describir una visión donde "las personas estarán rodeadas de interfaces inteligentes e intuitivas embebidas en objetos cotidianos de nuestro alrededor y un entorno que reconocen y responden a la presencia de individuos de manera invisible [4]. Es así que el objetivo de la AmI consiste en la creación de espacios donde los usuarios interaccionen de forma natural y sin esfuerzo con los diferentes sistemas.

Uno de los aspectos más importantes de la Inteligencia ambiental son las aplicaciones sensibles al contexto, es por ello que es necesario entender a que nos referimos con este concepto. Existen varias definiciones: La primera fue dada en [5], la cual restringía a estas a ser simplemente aplicaciones que eran informadas sobre el contexto para que se adaptaran a este. Otras definiciones como la de [6] definían a la computación sensible al contexto como la habilidad que poseen los dispositivos de detectar, sensar, interpretar y responder a los aspectos locales al ambiente de un usuario. Por su parte, Dey [7] define la noción de context-awareness como el "uso del contexto para automatizar un sistema de software, modificar su interface y proveer la máxima flexibilidad en términos de servicios".

Otra definición un poco más pasiva es la dada en [8], donde una aplicación sensible al contexto es aquella que pueda variar o adaptar dinámicamente su comportamiento en base al contexto. Salber [9] define que una aplicación sensible al contexto es aquella que tiene la capacidad de proporcionar la máxima flexibilidad de un servicio de cómputo basado en el censado del contexto en tiempo real.

Por último, en una forma más general pero a la vez concreta, en [10] Dey define a un sistema sensible al contexto como: "Un sistema (aplicación) es sensible al contexto si este hace uso de información del contexto para proveer información o servicios relevantes al usuario, donde la relevancia depende de actividad actual del usuario". Esta definición resulta ser la mayormente aceptada.

El auge de la Web 2.0 ha hecho que la industria incorpore las nuevas tecnologías que dieron lugar a su origen y se ajusten a los estándares emergentes. Mientras aun hoy no existe consenso sobre el alcance del término Web 2.0, O'Reilly ofrece una definición comúnmente aceptada, que incluye una gama de servicios mejorados, como ser los servicios web, wikis, blogs, bittorrents, y la sindicación de contenidos [11].

El crecimiento de la Web 2.0 también ha introducido un número de patrones de diseño y nuevos estilos arquitectónicos en el desarrollo

web. Una de las técnicas más destacadas consiste en la “mezcla” o mashup.

En [12] se presenta una definición precisa y completa la cual define a un mashup como una “Aplicación basada en Web que se crea mediante la combinación y procesamiento de los recursos en línea de terceros, que contribuyen con datos, forma de presentación o funcionalidad”. En esta definición, los recursos en línea de terceros se refieren a cualquier tipo de recursos disponibles en Internet, independientemente del formato. Como resultado, un mashup proporciona un nuevo recurso.

En términos generales, los mashup se crean usando el navegador web, “arrastrando y soltando” es decir, poniendo juntas aplicaciones de diferentes fuentes. Sin embargo, estas operaciones deben tener una infraestructura en el back-end para que de soporte al mashup, por ello en [13,14], se presenta una arquitectura de mashup que consta de tres componentes la cual se puede ver en la Figura 1. a) La API / proveedores de contenido. Estos son los proveedores de contenido (a veces heterogéneos) de los cuales se desea hacer el mashup. Para facilitar la recuperación de datos, los proveedores suelen dar acceso a sus contenidos a través de protocolos de Internet tales como REST, los Web Services o feeds RSS / Atom o utilizando screen scraping [15].b) El hosting del mashup. Es donde el mashup se encuentra alojado. Los mashups pueden utilizarse de manera similar a las aplicaciones Web tradicionales utilizando un servidor de contenido dinámico (Java Servlets, CGI, PHP o ASP). Alternativamente, el contenido del mashup se puede generar directamente en el navegador del cliente utilizando por ejemplo JavaScript o applets. c) El navegador Web de los consumidores. Es donde la solicitud se representa gráficamente y la interacción del usuario con el mashup se lleva a cabo.

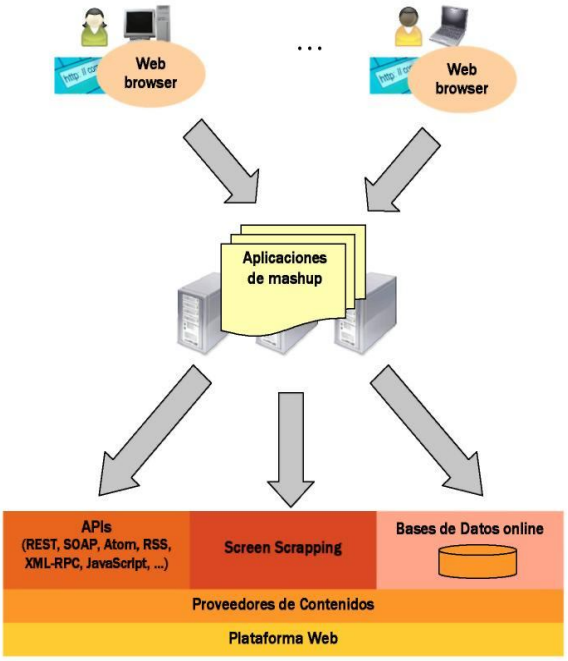

Figura 1. Arquitectura de Mashups

Líneas de investigación y desarrollo

En esta sección se analizarán dos proyectos de plataformas existentes para construir mashups sensibles al contexto:

1) El Caso del Proyecto Deusto Sentient Graffiti (DSG):El proyecto Deusto Sentient Graffiti [16, 17] busca la convergencia de la computación móvil y la computación ubicua con la web 2.0 para configurar lo que hoy día se llama Web Ubicua (UW). Esta última se puede definir como una infraestructura de Internet omnipresente en la que todos los objetos físicos son recursos accesibles a través de una URI. Estas URIs pueden proporcionar información y servicios que enriquezcan la experiencia de los usuarios en su contexto físico.

Su funcionamiento se basa en las anotaciones espontáneas o “Graffitis” (en formato de documento XML) que realiza una comunidad de usuarios sobre objetos, lugares o incluso

personas, con contenido web multimedia y/o servicios que pueden ser descubiertos y utilizados a su vez por otros usuarios móviles cuyos atributos contextuales coincidan con las de las anotaciones. La funcionalidad de DGS consiste en dos procesos principales: 1) Anotación de Grafitis: Los usuarios de los clientes DGS anotan objetos o regiones espaciales las cuales consistiendo de Descripciones, palabras clave y atributos contextuales y 2) descubrimiento y consumo de Grafitis, cuando los usuarios, pueden moverse virtual o físicamente a través de un ambiente navegando y consumiendo las anotaciones disponibles

DGS puede verse como mashup sensible al contexto dado que los clientes de DGS consiguen combinar la información geográfica en forma de mapas (ejemplo Google Maps) y la información suministrada por el servidor DSG en las descripciones.

La arquitectura de DGS presenta los siguientes componentes (Figura 2): a) Administrador de Grafiti y Web Server: administra todos los grafitis virtuales publicados b) Disparador y Notificador de Grafitis: Es un motor de inferencia basado en reglas que deduce los conjuntos de anotaciones activas que tienen notificarse a los usuarios c) Motor de Consultas de Grafitis: Este componente es responsable del procesamiento de consultas y de la recuperación de la anotaciones sensibles al contexto d) Monitores de Cambio de Contexto: Se ejecutan en el dispositivo móvil o navegador web vigilando el contexto del usuario, y e) Reproductor de Grafitis: Muestra grafitis virtuales en el dispositivo del usuario.

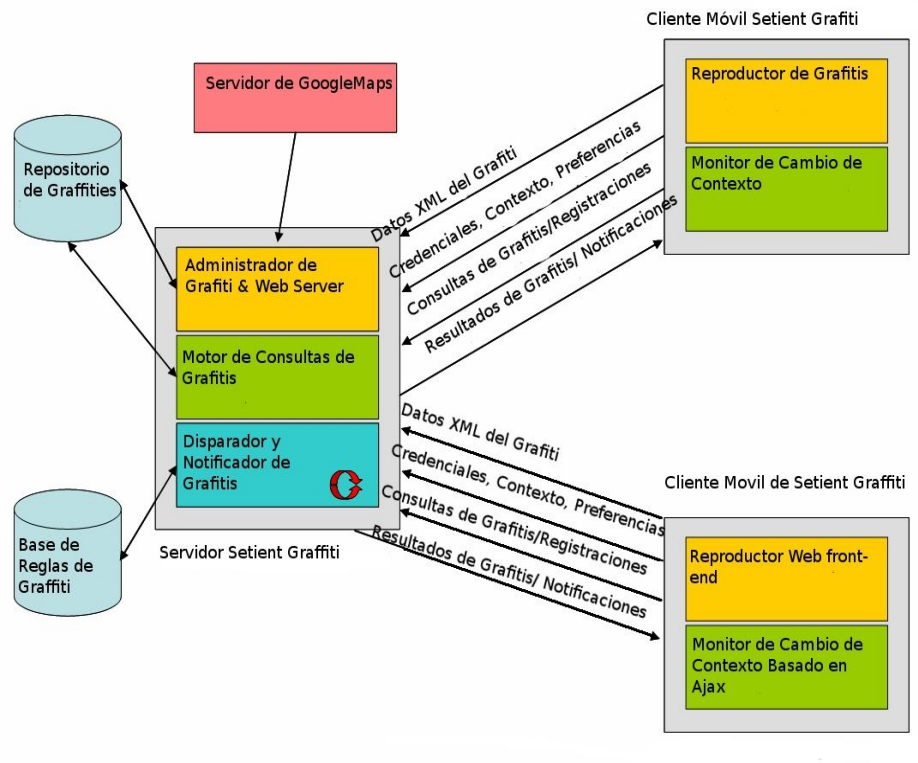

Figura 2. Componentes de DSG

Para su implementación DGS utiliza una sintaxis llamada GraffitiXML que se usa para especificar contenido. Como todo sistema basado en Web 2.0, como Google maps o Flickr, DGS provee una API basada en HTTP que permite a programadores de terceras partes desarrollar aplicaciones que se beneficien de su funcionalidad.

La implementación del servidor DGS está basada en Java y se ejecuta sobre el servidor de aplicaciones Tomcat utilizando JAVA EE. Además utiliza los marcos de trabajo, Tapesty, Hibernate, y Java Expert Systems Shell y como base de datos MySQL.

2) El Caso del Proyecto TELAR: Los autores de TELAR para la plataforma en [18] y que luego amplían en [19] se pueden resumir de la siguiente manera: a) La adaptación debe estar basada en sensores que se integran en el dispositivo móvil o conectado localmente. B) Los mashups deben ser centrados en el usuario, es decir, el usuario de un dispositivo móvil debe beneficiarse de las aplicaciones web híbridas, en lugar de otros usuarios distantes o los proveedores de servicios, c) Los mashups deben ser visibles con el navegador web del dispositivo móvil. Esto impide una solución nativa para el dispositivo móvil y tiene influencia potencial en el rendimiento, d) Debe existir una versión no sensible al contexto del mashup en caso de que la información de contexto no esté disponible. Esto permite que se puedan ver los mashups en los dispositivos móviles equipados con sensores, así como en equipos que no los posean e) Debe existir la posibilidad de que múltiples fuentes de datos con formatos de datos distintos e interfaces arbitrarias puedan integrarse en los mashups.

La base para la integración es un mapa provisto por un servicio de mapas en línea (Google Maps) y de la ubicación actual del usuario (GPS). Se integran diferentes proveedores de puntos de de interés (POIs). Los POIs se muestran en el mapa con diferentes símbolos. A medida que el usuario se mueve, el mapa se mantiene centrado en su posición y se muestra un rastro de su movimiento si el usuario así lo desea. Por el lado del cliente, la especificación Delivery Context Client Interfaces (DCCI) se utiliza para integrar la información de contexto de local obtenida a través de sensores con el navegador web móvil, que adapta el mashup a la localización actual del usuario.

En la Arquitectura presentada en la Figura 4 se puede observar la arquitectura del sistema TELAR, que de la misma manera que una aplicación mashup típica tiene tres niveles. El mashup se visualiza en la capa del cliente. El navegador carga la página web del mashup y ejecuta el código Javascript (AJAX) en el cliente. Esta página del mashup se carga desde el servidor de Mashup, que se encuentra en Internet en la capa servidor. Los datos ofrecidos por terceros proveedores que se utilizan se distribuyen por toda la web, estos se encuentran en la Capa de Proveedores de datos. Los mapas se cargan desde un servicio de mapas, que junto con los proveedores de datos, constituyen la capa de datos. Esta última capa se encuentra fuera de los límites de la organización del mashup.

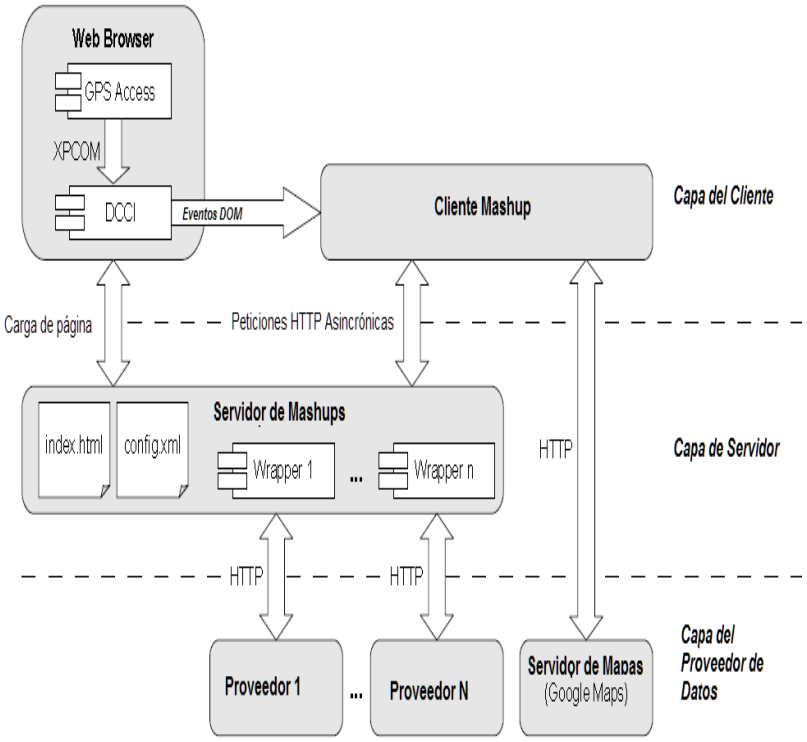

Figura 3. Componentes de TELAR

El cliente mashup utiliza lo desplegado en el servidor de mashup y estos pueden ser configurados/ parametrizados pero no necesitan ser programados por los usuarios finales. El cliente mashup lee asincrónicamente la configuración cuando la página del mashup se carga. Luego el cliente mashup construye la interfaz de usuario, esta muestra el mapa y visualiza los datos POIs de los proveedores de datos.

Una capa de normalización se utiliza lograr una interfaz consistente para la recuperación de datos utilizando wrappers, ya que este enfoque ofrece el más alto grado de flexibilidad. Para otros escenarios con menos diversidad, se pueden utilizar enfoques de normalización automática (por ejemplo, proveedores de datos que proporcionan canales RSS, como el usado en Yahoo Pipes [20]).

Tanto la información de contexto, como la ubicación del usuario, se integran al mashup, ampliando el navegador web. Para ello se utilizan dos componentes adicionales: el módulo DCCI y el módulo de acceso GPS. El módulo que implementa la especificación DCCI constituye la interfaz para proporcionar datos de contexto a las páginas web. El cliente mashup se registra como un detector de eventos en el módulo DCCI y este es notificado cada vez que se produce un cambio en la localización del usuario. El módulo de acceso GPS se conecta al dispositivo GPS y este pasa la información de ubicación necesaria al módulo DCCI.

Para implementar el cliente se utilizó el marco de trabajo Google Web Toolkit (GWT). Este dio la posibilidad de desarrollar un cliente mashup complejo en Java utilizando herramientas profesionales. Del lado del servidor, TELAR

requiere una aplicación servidor desde la cual el mashup se cargue y donde residan los wrappers de los proveedores de datos.

Resultados

Si bien existen herramientas para construir mashups sensibles al contexto en entornos de AmI estas son incipientes y todavía están en fase de prototipo. Aún no se cuenta con herramientas con la potencia y facilidad de uso de Yahoo Pipes para escenarios AmI.

Las tecnologías utilizadas, la forma de interacción con los usuarios, y de adquisición de conocimiento de este nuevo tipo de mashup, no están estandarizadas y existen múltiples estrategias y plataformas (como las mencionadas en la sección 4) para su construcción por lo que la mayoría de estas propuestas planteen una solución práctica para el escenario específico en cual son aplicadas.

Esto hace difícil integrar datos obtenidos desde sensores y utilizarlos para adaptar el contenido o el comportamiento de los mashups (y de cualquier aplicación web) ya que no existe un estándar de cómo capturar, representar y utilizar la información de contexto en las aplicaciones web.

Lo comentado anteriormente da lugar la necesidad de patrones, tanto para la adquisición de datos desde sensores (e información derivada) como en las estrategias para que estos datos sean aprovechados por los sistemas mashups de forma segura y que no comprometan la privacidad. Detallar sintéticamente los ejes del tema que se están investigando.

Como líneas futuras se proponen dos pilares, uno desde el punto de vista de las aplicaciones para la creación de mashups sensibles al contexto, se plantea la necesidad de contar con una herramienta del tipo de Pipes de Yahoo, pero que además de fuentes de datos RSS, APIS, y cajas de texto de entrada de datos (user input) pueda incorporar información de contexto provista por los sensores (context input) del dispositivo móvil del usuario de manera automática.

Siguiendo con esta línea como el segundo pilar sería interesante contar con un navegador web sensible al contexto, que de soporte a la AmI. Este Navegador de Web Ubicua (navegador sensible al contexto), se lo podría plantear como una evolución de un navegador tradicional como Mozilla Firefox o una extensión para los ya existentes. De esta forma se podría realizar una navegación de Ambientes Inteligentes. Esta extensión o mejora permitiría recabar datos de los sensores de dispositivos móviles de forma transparente para el usuario y las aplicaciones mashups e independientemente del hardware o tecnología subyacente, En este sentido se esta trabajando con HTML5 y formatos interoperables como ser REST y RSS y abstraer WSN (Redes de Sensores Inalámbricos por sus siglas en inglés) de la publicación de datos que ellas generan para lograr el objetivo.

Formación de Recursos Humanos

El equipo de trabajo se encuentra formado por un Doctor en Ciencias Informáticas, Un Doctorando en Ingeniería Telemática, un auxiliar de investigación graduado, y ocho auxiliares (Resolución rectoral 21/I/12) de investigación en período de realización de trabajos finales de grado. El número de tesinas de grado en curso con proyecto aprobado es tres y el número de trabajo de especialidad en curso con proyecto aprobado es uno. Los proyectos de grado se titulan “Plataforma para la publicación de datos de Redes de Sensores Inalámbricos, orientada a la visión de la Internet de las Cosas, Ambientes

Inteligentes y Mashups", "Diseño de un prototipo para monitoreo eficiente de iluminación basado en WSN utilizando HTML5" y "Contribución a la Gestión de Residuos Domiciliarios como una Aplicación en Ciudades Inteligentes". El Trabajo de Especialidad se titula "Plataformas para la creación de Mashups Sensibles al Contexto en Entornos de Inteligencia Ambiental"

Referencias

[1] Weiser M, The Computer for the 21st Century. Scientific American, 265(3):66–75, September 1991

[2] Weiser, M.. Some computer science issues in Ubiquitous Computing. Communications of ACM, 36(7), 75-84 1993

[3] Sosa, Eduardo O., Contribuciones al establecimiento de una red global de Sensores Inalámbricos. Tesis Doctoral. s.l.: Universidad Nacional de La Plata, Junio 17, 2011.

[4] Ahola J, Ambient Intelligence, ERCIM News, No 47, October 2001.

[5] Schilit, B., Theimer, M. Disseminating Active Map Information to Mobile Hosts. IEEE Network, 8(5) (1994) 22-32

[6] Pascoe, J. Adding Generic Contextual Capabilities to Wearable Computers. 2nd International Symposium on Wearable Computers (1998) 92-99

[7] Dey, A.K. Context-Aware Computing: The CyberDesk Project. AAAI 1998 Spring Symposium on Intelligent Environments, Technical Report SS-98-02 (1998) 51-54

[8] Gerd Kortuem, Zary Segall, and Martin Bauer. Context-aware, adaptive wearable computers as remote interfaces to 'intelligent' environments. In ISWC, pages 58–65, 1998.

[9] Salber, D., Dey, A.K., Abowd, G.D. Ubiquitous Computing: Defining an HCI Research Agenda for an Emerging Interaction Paradigm. Georgia Tech GVU Technical Report GIT-GVU-98-01 (1998)

[10] Anind Kumar Dey. Providing architectural support for building context-aware applications. PhD thesis, Georgia Institute of Technology, 2000. Director-Gregory D. Abowd.

[11] T. O'Reilly, "What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software", O'Reilly Media Inc., September 2005.

[12] A. Koschmider, V. Torres, V. Pelechano: Elucidating the Mashup Hype: Definition, Challenges, Methodical Guide and Tools for Mashups. 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009).

[13] Duane Merrill. Mashups: The new breed of Web app--Anintroduction to mashups. http://www.ibm.com/developerworks/xml/library/x-mashups.html

[14] Xuanzhe Liu; Yi Hui; Wei Sun; Haiqi Liang, "Towards Service Composition Based on Mashup," Services, 2007 IEEE Congress on , vol., no., pp.332-339, 9-13 July 2007

[15] Sitio de Internet: Screen Scrapper, http://www.screen-scraper.com/

[16] López de Ipiña D., Vázquez J.I. and Abaitua J., Context-Aware Mobile Mash-up for Ubiquitous Web, Puertollano, Spain, ISBN: 84-6901744-6, pp. 19-34, November 2006

[17] D. Lopez-De-Ipina, J. I. Vazquez, J. Abaitua. "A context-aware mobile mashup platform for ubiquitous web". Intelligent Environments, 2007. IE 07. 3rd IET International Conference on (2007), pp. 116-123.

[18] Andreas Brodt , Daniela Nicklas, The TELAR mobile mashup platform for Nokia internet tablets, Proceedings of the 11th international conference on Extending database technology: Advances in

database technology, March 25-29, 2008, Nantes, France

[19] A. Brodt, D. Nicklas, S. Sathish, and B. Mitschang. Contextaware mashups for mobile devices. In Web Engineering (WISE '08), pages 280–291. Springer-Verlag, 2008.

[20]Sitio de Internet: Yahoo Pipes, http://pipes.yahoo.com/pipes/