JXME: Una plataforma robusta para el desarrollo de aplicaciones P2P en dispositivos móviles

Francisco Orlando Martínez Pabón*,
Ricardo Alberto Camacho Gómez**,
Luis Ernesto García Martínez***,
Oscar Mauricio Caicedo Rendón****,
Javier Alexander Hurtado Guaca*****

Resumen

Paralelamente a la gran difusión de las tecnologías P2P, los dispositivos móviles, principalmente teléfonos celulares, PDAs y smartphones, se convierten en la gran ola de dispositivos conectados a Internet de nueva generación. Siguiendo esta tendencia, Sun Microsystems lanzó el proyecto JXTA, un conjunto de APIs y protocolos para el desarrollo de aplicaciones P2P, y posteriormente una especialización del mismo para dispositivos móviles conocida como JXME.

Este artículo describe el trabajo realizado por el Grupo de Interés en el Desarrollo de Aplicaciones Móviles e Inalámbricas – W@PColombia, perteneciente al Grupo de Ingeniería Telemática de la Universidad del Cauca, acerca del desarrollo de aplicaciones P2P para dispositivos móviles con JXME. A través de un prototipo de validación conocido como Punto de Encuentro Virtual Móvil se logró comprobar la viabilidad de este tipo de aplicaciones en la redes de 2.5G y se realizaron aportes importantes al trabajo desarrollado por la comunidad JXTA.

Palabras claves: Anuncio, grupo de peers, J2ME, mensaje, P2P, tubería, repetidor.

Fecha de recepción: 2 de febrero de 2006
Fecha de aceptación: 12 de diciembre de 2006

* Ingeniero en Electrónica y Telecomunicaciones. Grupo de Ingeniería Telemática, Universidad del Cauca. *fomarti@unicauca.edu.co*

** Ingeniero en Electrónica y Telecomunicaciones, Universidad del Cauca. *rcamacho@unicauca.edu.co*

*** Ingeniero en Electrónica y Telecomunicaciones, Universidad del Cauca. *lgarcia@unicauca.edu.co*

**** Ingeniero en Electrónica y Telecomunicaciones. Magister en Ingeniería, Área Telemática. Grupo de Ingeniería Telemática, Universidad del Cauca. *omcaicedo@unicauca.edu.co*

***** Ingeniero en Electrónica y Telecomunicaciones. Especialista en Redes y Servicios Telemáticos. Grupo de Ingeniería Telemática, Universidad del Cauca. *javhur@unicauca.edu.co*
Dirección: Universidad del Cauca, Departamento de Ingeniería Electrónica, Calle 5 N° 4-70, Popayán (Colombia).

Origen de los apoyos recibidos para la realización del proyecto: Grupo de Ingeniería Telemática, Departamento de Telemática. Recursos propios. Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca.

Abstract

About the growth of P2P technologies, the mobile devices, mainly cell phones, PDAs y smartphones have been the representative mass of Internet connected devices of the new generation. According to this trend, Sun Microsystems released the JXTA project, a set of APIs and protocols for P2P applications development, and later it released JXME, a JXTA specialization for mobile devices.

This article introduces a description about the work of the Wireless and Mobile Applications Interest Group – W@PColombia as part of the Telematics Engineering Group from University of Cauca, about P2P applications development for mobile devices with JXME. Throughout a validation prototype, it was possible to test the mobile P2P applications viability in 2.5G networks and important contributions to JXTA development community were made.

Key words: Advertisement, Peer group, J2ME, Message, P2P, Pipe, Relay.

INTRODUCCIÓN

En la era post-PC, los usuarios requieren tener acceso a la información a cualquier hora y en cualquier lugar, lo cual implica que los terminales conectados a Internet ya no son sólo computadores personales de ubicación fija, sino dispositivos móviles como teléfonos celulares, smartphones, computadores portátiles y PDAs (*Personal Digital Assistant*), que han generado una gran ola conocida como "Internet Móvil".

Actualmente, la tecnología de computación distribuida o descentralizada *Peer-to-Peer* (P2P) [1] [2] tiene un futuro promisorio como el mecanismo para superar las deficiencias en el acceso a la información que demandan los usuarios móviles (mensajería instantánea, distribución de archivos, juegos colaborativos, etc). En el marco de las redes P2P, Sun Microsystems creó el proyecto de fuente abierta (*open source*) JXTA [3], el cual básicamente define un conjunto de protocolos para estandarizar el desarrollo de aplicaciones sobre este tipo de redes. En la misma vía, y bajo la misma filosofía, Sun creó el proyecto JXME [4], una versión de JXTA optimizada para dispositivos móviles Java 2 MicroEdition (J2ME) [5]. El proyecto JXME propone una arquitectura y un conjunto de APIs que facilitan el acceso transparente a redes P2P por parte de los dispositivos móviles, teniendo en cuenta sus limitaciones de memoria y capacidad de procesamiento. Sin embargo, a través de un estudio de las APIs y la implementación de un prototipo de prueba basado en JXME se comprobó que persisten algunas falencias relacionadas con la operación de terminales móviles en una red P2P.

Este artículo describe las características principales de los proyectos JXTA y JXME, y se valida su implementación a través de la descripción de un prototipo de mensajería instantánea móvil denominado "Punto de Encuentro Virtual Móvil" y analiza los fallos encontrados en la implementación proporcionada por el proyecto JXME; igualmente, se describen las soluciones y aportes realizados a la comunidad JXTA. Los resultados del proyecto presentado en este artículo se enmarcan dentro del trabajo que desarrolla el Grupo de Ingeniería Telemática (GIT) de la Universidad del Cauca, a través de la línea de interés en el Desarrollo de Aplicaciones Móviles e Inalámbricas (Grupo W@PColombia).

Este artículo se ha dividido en las siguientes secciones: en la sección I se realiza una breve descripción del proyecto JXTA; en la II se describe el proyecto JXME; en la III se presenta el Punto de Encuentro Virtual Móvil; en la IV se hace referencia a los aportes realizados al proyecto JXME, y finalmente se establecen las conclusiones más importantes sobre el trabajo realizado.

1. EL PROYECTO JXTA

Desafortunadamente, en la actualidad las aplicaciones P2P tienden a usar protocolos propietarios e incompatibles por naturaleza; cada red forma una comunidad completamente cerrada, independiente de las otras redes e incapaz de impulsar sus servicios. Hasta el momento, la emocionante exploración de aplicación de la tecnología P2P ha opacado la importancia de la interoperabilidad y reutilización de *software* [6]. Para convertir a P2P en una tecnología madura para la construcción de soluciones se requiere de APIs unificadas que faciliten el trabajo a los desarrolladores y un lenguaje común o protocolos que les permita a los participantes (*peers*) comunicarse y realizar diversas tareas sobre la red P2P. Teniendo en cuenta estas necesidades, Sun Microsystems creó el Proyecto JXTA (pronunciado yuxtapous o yuxta) con el ánimo de crear un común denominador en el ambiente de las redes P2P [7]. En su interior, JXTA es simplemente un conjunto de especificaciones protocolares sobre las cuales se pretende soportar una extensa gama de aplicaciones; de esta forma, quien desee producir una nueva aplicación se ahorra la dificultad de diseñar sus propios protocolos para manejar el núcleo de funciones de una comunicación P2P.

El nombre JXTA se deriva de la palabra "juxtapose" (yuxtapuesto), que significa colocar dos entidades en paralelo o en proximidad, una al lado de la otra. Escogiendo este nombre, el equipo de desarrolladores de Sun reconoció que las soluciones P2P siempre existirían junto a las actuales soluciones cliente/servidor en lugar de reemplazarlas completamente [8]. El Proyecto JXTA no ha sido diseñado para sustituir las tecnologías y los modelos de computación

actuales, sino para ampliarlos con el fin de satisfacer un nuevo conjunto de necesidades.

Los protocolos JXTA están diseñados para ser independientes del lenguaje de programación y de los protocolos de transporte. Estos pueden ser implementados en Java, C/C++, Perl, y en muchos otros lenguajes de programación [9]. Dichos protocolos se pueden basar en TCP/IP, HTTP, Bluetooth, Home PNA u otros protocolos de transporte. Adicionalmente, JXTA define una arquitectura basada en capas, diferenciando claramente cada uno de los componentes que intervienen en una comunicación P2P. Una descripción detallada acerca de la arquitectura JXTA se encuentra en [10] [11].

1.1 Operación de una red JXTA

La red JXTA consiste en una serie de nodos interconectados, más conocidos como *peers*. Sensores, teléfonos móviles, PDAs o computadores personales pueden ser peers de una red JXTA. Cada *peer* opera de manera asíncrona e independiente de otros *peers* y posee un identificador único, conocido como identificador de *peer* (*PeerID*). Dichos *peers* se organizan en grupos de *peers* (*peer groups*), los cuales proveen un conjunto de servicios en común. Cada grupo de *peers* puede establecer sus propias políticas de acceso y privacidad, desde abierta (cualquiera puede añadirse al grupo) hasta altamente segura y protegida (solicitud de suficientes credenciales para el ingreso al grupo). Por defecto, el primer grupo instanciado y al cual pertenecen todos los *peers* es el *NetPeerGroup* [12].

Para dar a conocer los servicios que ofrecen, los *peers* JXTA utilizan un mecanismo conocido como anuncios (*advertisements*), que consisten en documentos XML que permiten a otros *peers* en la red JXTA conectarse e interactuar con los servicios que ofrece un *peer* en particular. Para transferir los anuncios, los *peers* JXTA usan un medio abstracto conocido como tubería (*pipe*); las tuberías son un mecanismo de transferencia de mensajes asíncrono y unidireccional usado para prestar el servicio de comunicación a los *peers*; cada tubería está asociada a terminales (*endpoints*) específicos, tales como un puerto TCP y una dirección IP asociada. Al representar canales virtuales de comunicación, las tuberías pueden conectar *peers* que no tienen un enlace físico directo; en este caso, uno o más *peers* intermedios deben intervenir como repetidores de los mensajes que se desean intercambiar entre dos puntos extremos de la tubería. Teniendo en cuenta las características de comunicación mencionadas anteriormente, las tuberías ofrecen dos modos de comunicación: punto a punto y punto a multipunto. En la figura 1 se observan los principales componentes que intervienen en la operación de una red JXTA.

2. EL PROYECTO JXME (JXTA for J2ME)

Una nueva generación de dispositivos inalámbricos, entre los que se encuentran los teléfonos móviles, las PDAs y los smartphones, han incrementado considerablemente su demanda en los últimos años, y el desarrollo de aplicaciones para este tipo de dispositivos mantiene una proyección exponencial.

Por supuesto, las tecnologías P2P no han sido ajenas a este fenómeno y lideran una tendencia hacia entornos más dinámicos y colaborativos entre los usuarios de estos dispositivos inalámbricos [13].

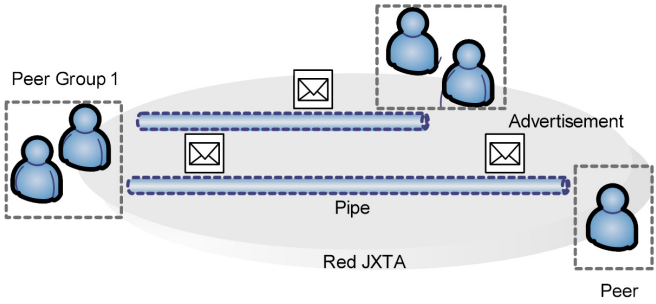

Figura 1. Componentes de una red JXTA

El proyecto JXME es la respuesta de JXTA a esta tendencia y busca facilitar la compatibilidad entre la red JXTA y los dispositivos móviles con soporte J2ME enmarcados dentro de la configuración CLDC 1.0 (*Connected Limited Device Configuration*) y el perfil MIDP 1.0/2.0 (*Mobile Information Device Profile*), los cuales definen un conjunto de APIs para el manejo de tareas generales, interfaz de usuario, *networking* y persistencia requeridos en el desarrollo de aplicaciones para este tipo de dispositivos. Gracias a JXME, cualquier dispositivo MIDP puede participar en redes P2P con otros dispositivos MIDP y además usar el soporte que ofrece JXTA para participar en actividades P2P con *peers* ubicados en computadores de escritorio, estaciones de trabajo y servidores.

JXME implementa las capacidades de búsqueda de *peers*, ingreso a grupos *peer* y localización de tuberías. La tecnología JXME ha sido probada en dispositivos Palm OS, teléfonos móviles con soporte MIDP 2.0/1.0 CLDC 1.0 y en diversos simuladores J2ME, lo cual facilita la verificación de prototipos desarrollados con esta tecnología y la puesta en funcionamiento en entornos reales [14].

2.1. Operación de JXME

Aunque J2ME define un conjunto de APIs que permite establecer la viabilidad del desarrollo de *peers* móviles, aún este tipo de dispositivos enfrenta una serie de limitaciones asociadas a la capacidad de procesamiento, memoria y mecanismos para la entrada de información (teclados numéricos) que se deben tener en cuenta. Para conectar dispositivos J2ME a redes JXTA conviviendo con las limitaciones mencionadas, los diseñadores del proyecto JXME decidieron llevar gran parte de la carga de las aplicaciones JXTA basadas en J2ME sobre una clase de *peers* más capacitados para manejar los volúmenes de tráfico que se generan en la red JXTA; de esta forma, los *peers* JXME dan una funcionalidad reducida comparada con la de los *peers* de escritorio. Para superar entonces las limitaciones de un dispositivo J2ME se hace uso de *peers* especiales conocidos como repetidores JXTA, los cuales actúan como repetidores de mensajes y además asumen el soporte de la mayor parte del procesamiento de mensajes o anuncios. Un *peer* basado en J2ME, en conjunto con un repetidor JXTA, es funcionalmente equivalente a un *peer* JXTA normal. Por lo tanto, un *peer* J2ME actuará como un dispositivo terminal, fijado en el perímetro de una red JXTA. La figura 2 ilustra cómo un repetidor ayuda a los dispositivos basados en J2ME a interactuar con una red JXTA [15].

Cuando un *peer* J2ME envía un mensaje, lo encapsula en una petición HTTP; el repetidor JXTA analiza cada par nombre-valor en la petición recibida, produce mensajes XML de acuerdo con el formato JXTA y retransmite los mensajes hacia la red para realizar las funciones P2P determinadas por la estructura y contenido del mensaje.

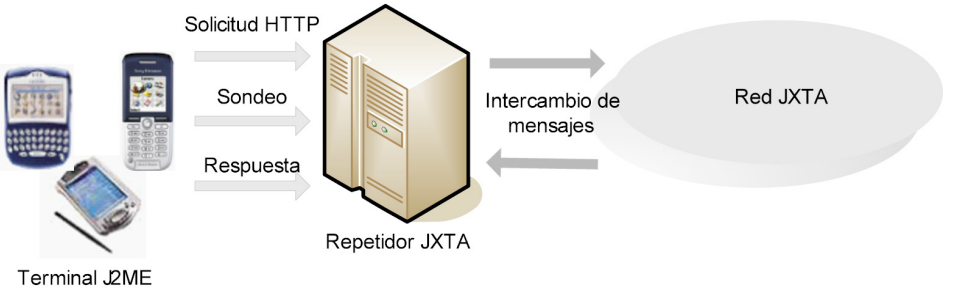

Figura 2. Interacción entre dispositivos J2ME con la red JXTA a través de un repetidor

En el caso opuesto, cuando un repetidor JXTA recibe un mensaje desde la red JXTA y éste tiene como destino un *peer* J2ME, analiza el formato del men-

saje entrante y produce la respuesta HTTP correspondiente. Sin embargo, los dispositivos J2ME que soportan la versión MIDP 1.0 no tienen la funcionalidad de un servidor HTTP y, por ende, no pueden abrir puertos para escuchar peticiones entrantes; de acuerdo con estas limitaciones, un repetidor JXTA no tiene medios para enviar la respuesta HTTP de regreso al peer J2ME. Por esta razón, el repetidor debe esperar que el peer J2ME envíe una petición de sondeo, en respuesta a la solicitud enviada previamente. Cuando esto pasa, el repetidor JXTA envía la respuesta HTTP de vuelta al *peer* J2ME.

3. EL PUNTO DE ENCUENTRO VIRTUAL P2P

El Punto de Encuentro Virtual P2P es una aplicación basada en el conjunto de protocolos JXTA para terminales móviles J2ME que soportan los perfiles MIDP 1.0 o 2.0. Aunque existen otros sistemas de mensajería instantánea para dispositivos móviles J2ME, como clientes IRC (JimIRC [16]), clientes ICQ (Jimm [17]), o sistemas basados en otro conjunto de protocolos, como Jabber [18] (Mobber [19], un comunicador móvil basado en Jabber para teléfonos con GPRS), las implementaciones de clientes JXME aún no son muy difundidas.

El Punto de Encuentro Virtual P2P pretende facilitar a los usuarios de telefonía móvil el intercambio de información de interés e interactuar en un ambiente distribuido P2P. En términos generales, el Punto de Encuentro es un sistema de mensajería instantánea en el cual los usuarios pueden gestionar comunidades especializadas (grupos *peer*) en determinados temas (deportes, tecnología, amistades, entre otros), crearlas, unirse a las mismas e intercambiar información personal de cada usuario especificada en un perfil configurado y almacenado previamente en cada terminal (ver figura 3.1a).

Por defecto existen comunidades base a las cuales los usuarios pueden ingresar cuando deseen mientras se encuentren conectados a la red JXTA (ver figura 3.1b). El sistema permite a los usuarios saber cuáles *peers* están activos en un momento dado, y también anunciar la llegada o la salida de un *peer* a un grupo mediante la difusión multicast de mensajes informativos de este tipo de eventos (ver figura 3.1c). Por otra parte, los *peers* una vez conectados a un grupo tienen la posibilidad de enviar mensajes a todos los miembros o a un solo *peer* en particular de manera privada. Teniendo en cuenta que los medios de escritura de los dispositivos móviles no son muy cómodos y ágiles, se ha implementado también una lista de palabras con las expresiones más utilizadas por los usuarios, facilitando de esta manera la escritura desde los terminales JXME (ver figura3.1d). Además, se tiene a disposición un conjunto de *Emot-icons* que abrevian y hacen más agradable las sesiones de conversación en el Punto de Encuentro Virtual (ver figura 3.1e).

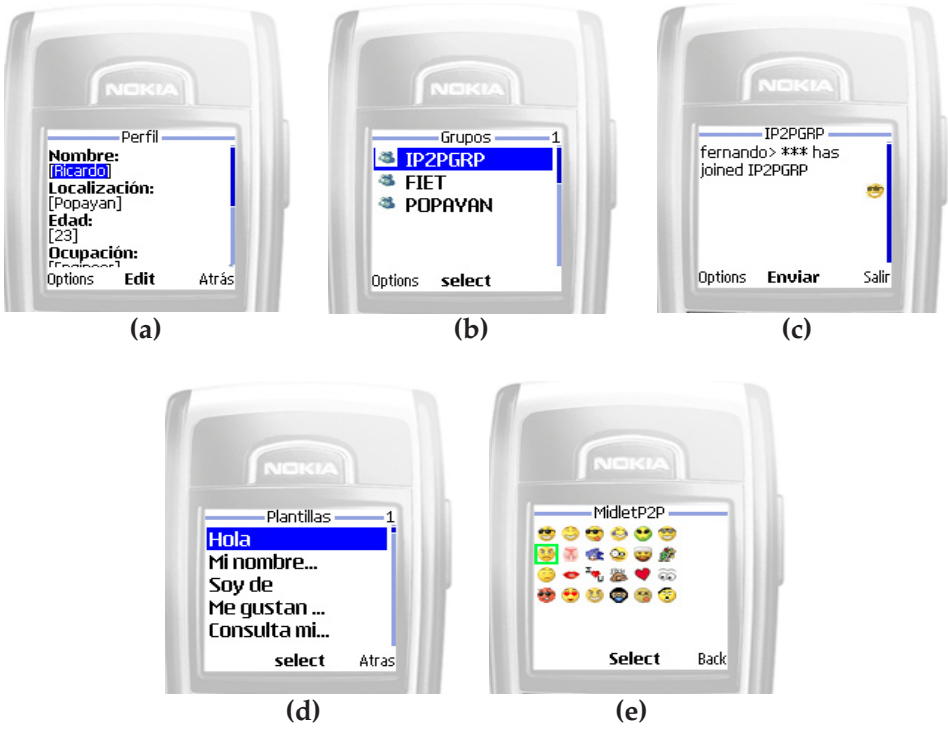

Figura 3. Interfaces gráficas de usuario del Punto de Encuentro Virtual en diversas etapas

Las pruebas del Punto de Encuentro Virtual P2P se realizaron con teléfonos móviles Nokia 3595, utilizando el servicio GPRS de la red de Colombia Móvil. De acuerdo con los resultados obtenidos, los tiempos de latencia fueron tolerables fijando un tiempo de sondeo desde los dispositivos móviles al repetidor de 1 segundo (ver tabla 1).

Tabla 1
Resultados de tráfico en pruebas de aplicaciones JXME sobre una red GSM/GPRS

Resultados de Tráfico Mensajería Instantánea (Punto de Encuentro)	
Tiempo Máximo de establecimiento de Conexión (Seg.)	50
Retardo máximo entre el envío y recepción de un mensaje (Seg.)	15
Retardo promedio entre el envío y recepción de un mensaje (Seg.)	7
Volumen de datos promedio en 6 minutos (Kbytes)	Enviado: 45,806KB Recibido: 28,235KB Total: 74,041KB

Adicionalmente, se obtuvieron algunos datos relacionados con el intercambio de información entre los dispositivos móviles y el repetidor utilizando el SDK Series 40 Developer Platform 2.0, el cual incorpora un analizador de tráfico muy útil que permite simular gran parte de las características soportadas por los dispositivos móviles con soporte J2ME más populares en Colombia (Nokia 6230, Nokia 3595, Sony Ericsson T610/T616, entre otros) (ver figura 4).

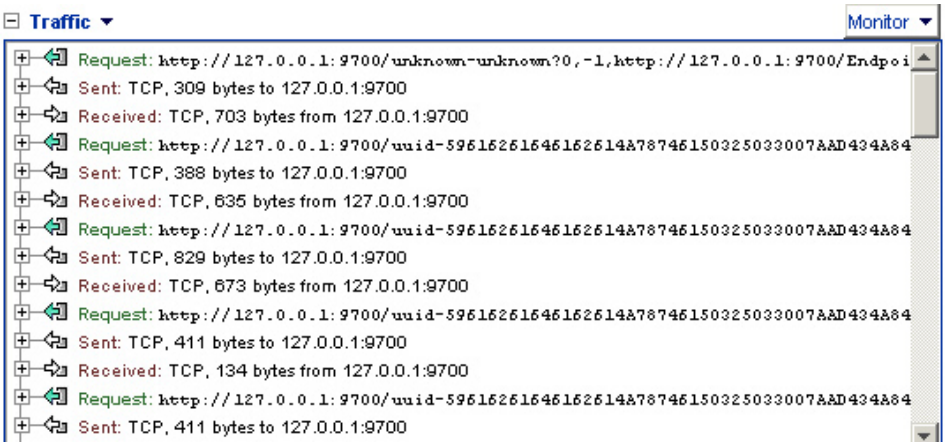

Figura 4. Analizador de tráfico Series 40 Developer Platform 2.0 SDK

La tabla 2 resume los resultados del análisis de tráfico en ambos sentidos (dispositivo móvil - repetidor y repetidor - dispositivo móvil) generado por una sesión de mensajería instantánea en cada etapa específica, de acuerdo con los lineamientos de operación de JXME:

- En la fase de *establecimiento de la conexión* se solicita un *peerId* y un *pipeId* para el móvil y por otra parte se descarga la lista de peer groups disponibles a los cuales el móvil puede unirse en el momento de su conexión.

- El *ingreso a un Peer group* disponible se realiza una vez que la conexión con el repetidor ha sido establecida.

- En la fase de *conversación* existen dos casos para analizar: Se sondea el repetidor en espera de mensajes entrantes (no hay mensajes para enviar o recibir y se envían mensajes vacíos para mantener la conexión con el repetidor), o bien en el sondeo se envían o reciben datos de un mensaje escrito. En el primer caso se tomó un tiempo de sondeo de referencia de 10 segundos y en el segundo caso se tomó como referencia un promedio de 30 caracteres por mensaje.

Tabla 2
Análisis de tráfico generado en los repetidores

Fase	Tráfico Uplink (Móvil a Relay) [Kbytes]	Tráfico Downlink (Relay a Móvil) [Kbytes]
Establecimiento de Conexión	1,526	2,011
Ingreso a *peer group*	0,823	0,493
Sondeo sin Envío/Recepción de mensaje	0,411	0,134
Sondeo con Envío/Recepción de mensaje	0,974	1,215

Por otro lado, es importante destacar que los repetidores JXME implementan un mecanismo para minimizar los efectos de una posible congestión. A medida que los mensajes llegan al repetidor se colocan en una cola de espera y se van evacuando de acuerdo con la prioridad que tienen los paquetes. De esta manera, los paquetes que transmiten información más sensible a los retardo tendrán un nivel de prioridad más alto con respecto a los demás paquetes.

De acuerdo con los resultados obtenidos en las pruebas se demuestra la viabilidad técnica para implementar aplicaciones P2P de este tipo en redes de 2.5G, aunque su verdadera adopción depende en gran parte de una adecuada estrategia de mercado por parte de los operadores de telefonía móvil.

4. APORTES AL PROYECTO JXME

A lo largo del proceso de implementación del Punto de Encuentro Virtual P2P basado en las diferentes características que ofrece JXTA para dispositivos J2ME, se descubrieron algunos fallos en las operaciones relacionadas con la gestión de grupos y el manejo de tuberías en la implementación del repetidor, ofrecida por el proyecto JXME. Para cumplir a cabalidad con la funcionalidad del Punto de Encuentro Virtual se desarrolló la solución pertinente para cada fallo en mención. De esta forma, se realizó un gran aporte a toda la comunidad mundial de desarrolladores JXTA.

Antes de entrar en detalles sobre los fallos encontrados y las soluciones implementadas, se realizará una breve descripción de la estructura de los mensajes intercambiados entre los peers J2ME y el Repetidor para facilitar la comprensión de los aportes realizados al proyecto JXME.

4.1 **Interacción entre *peers* J2ME y el Repetidor**

Los *peers* JXTA en general, sin importar si son móviles o no, se comunican entre sí a través de mensajes. Debido a las limitaciones en la comunicación HTTP de los dispositivos J2ME MIDP 1.0 que fueron señaladas en la sección 2.1 y en general a otro tipo de limitaciones relacionadas con la baja capacidad de memoria para el procesamiento de mensajes XML, el Repetidor asume la responsabilidad de representar al *peer* móvil en la red P2P. Como se dijo anteriormente, la comunicación de los *peers* J2ME con el Repetidor se realiza a través de peticiones HTTP sucesivas, en un procedimiento conocido como "polling" o sondeo. Mediante este procedimiento, los *peers* J2ME envían y reciben mensajes JXTA para realizar diferentes operaciones en la red, como las que se relacionan a continuación:

- Creación de grupos de *peers* y tuberías
- Búsqueda de *peers*, grupos de *peers* y tuberías
- Ingreso a grupos de *peers*
- Cierre de tuberías.

Los mensajes JXME están conformados por unidades llamadas "Elementos" (ver figura 5), encargados de llevar información específica del mensaje, como el tipo y nombre del mensaje, entre otros. Cada elemento, a su vez, se compone de cuatro campos de información, que se relacionan a continuación:

- *Name*: Campo que describe el nombre del elemento.
- *Data*: Contiene los datos del elemento que son transportados a través de toda la red JXTA.
- *NameSpace*: Describe el *namespace* usado por el elemento, los mensajes JXTA usan el *namespace* "JXTA". Los mensajes en JXME usan un *namespace* privado. Si el *namespace* es *null*, se usa el *namespace* por defecto "".
- *MIME Type*: Describe el tipo MIME de los datos que lleva el mensaje. Si se pone *null* en este campo, automáticamente se asume el *mime type* por defecto que es *"application/octet-stream"*.

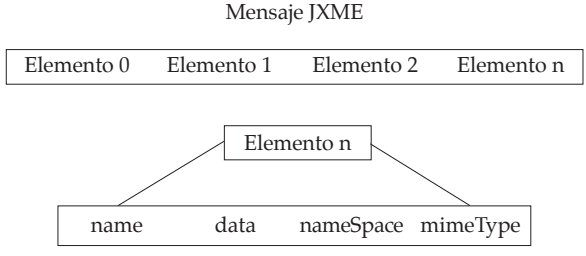

Figura 5. Estructura de los mensajes JXME

4.2 Ingreso a un grupo de *peers* (*Peer group*) diferente del universal NetPeerGroup

Cuando un *peer* se conecta a la red JXTA, automáticamente ingresa al grupo NetPeerGroup, grupo universal al que pertenecen todos los *peers* JXTA. En el desarrollo del proyecto se comprobó que desde este grupo la invocación de todas las operaciones disponibles para los *peers* J2ME se llevaban a cabo satisfactoriamente, sin embargo, al unirse a un grupo diferente ya no había respuesta de la red JXTA a los mensajes enviados desde el terminal J2ME; el *peer* permanecía sondeando el repetidor para enviar o recibir mensajes sin respuesta alguna. En la tabla 3 se resume el contenido del mensaje de petición que envía el peer J2ME para unirse a un nuevo grupo.

Tabla 3
Mensaje de petición de un peer J2ME para unirse a un nuevo grupo

Elemento N°	Nombre del elemento	Datos del elemento
0	Request	Join
1	Id	urn:JXTA:uuid-E25359ECA0524ED28C2EF85FE16C004E02
2	Arg	Null
3	Requested	2
4	EndpointDestinationAddress	HTTP://127.0.0.1:9700/EndpointService:uuid-E25359ECA0524ED28C2EF85FE16C004E02/urn:JXTA:uuid-DEADBEEFDEAFBABAFEEDBABE0000000E05/urn:JXTA:uuid-E25359ECA0524ED28C2EF85FE16C004E02
5	EndpointSourceAddress	JXTA://uuid-59616261646162614A787461503250334C7A8F4C38FA4CDD810C910A56CFF27603

Los elementos del mensaje se describen a continuación:

- *Request*: Indica la clase de petición realizada al Repetidor, en este caso *Join*, indicando una petición de ingreso a un grupo.
- *Id*: Especifica el identificador del grupo (*groupId*) al cual se desea ingresar.
- *Arg*: Identifica el *password* de acceso al grupo, si el grupo es privado. En el caso que el grupo sea público, este argumento es *null*.
- *Requested*: Este número especifica la cantidad de peticiones que fueron realizadas al Repetidor.
- *EndpointDestinationAddress*: Especifica la dirección del Repetidor al cual se hace la petición.

- *EndpointSourceAddress*: Especifica la dirección del *peer* que hace la petición de ingreso al grupo específico.

En la tabla 4 se puede observar el contenido del mensaje de respuesta recibido desde el repetidor:

Tabla 4

Mensaje de respuesta a la petición de un peer J2ME para unirse a un nuevo grupo

Elemento N°	Nombre del elemento	Datos del elemento
0	Response	Success
1	Requested	1
2	EndpointDestinationAddress	HTTP://127.0.0.1:9700/ EndpointService:JXTA-NetGroup/ urn:JXTA:uuid-DEADBEEFDEAFBABAFEEDBAB E0000000E05/ urn:JXTA:JXTA-NetGroup
3	EndpointSourceAddress	JXTA://uuid-59616261646162614A78746150325033 4C7A8F4C38FA4CDD810C910A56CFF27603

- *response*: Este elemento especifica el tipo de respuesta obtenido frente a la petición enviada. En este caso, se recibió "success", lo cual indica una respuesta exitosa.
- *requestId*: Este número especifica la cantidad de peticiones hechas al Repetidor.
- *EndpointDestinationAddress*: Especifica la dirección del Repetidor al cual se hace la petición.
- *EndpointSourceAddress*: Especifica la dirección del *peer* que hace la petición de ingreso al grupo de *peers*.

Posteriormente, para continuar con el registro de un *peer* en un nuevo grupo, es necesario crear una nueva instancia de la clase *PeerNetwork*, mediante la siguiente declaración: PeerNetwork.createInstance(String peerName, String peerGroupId); ésta instrucción hace que en cada nueva petición el elemento *EndpointDestinationAddress* cambie, de tal forma que ahora los mensajes van dirigidos al nuevo grupo al cual se ha unido el *peer*.

Este cambio es visible en la observación del elemento *EndpointDestinationAddress* que aparece en cada mensaje enviado al Repetidor, el cual se compone fundamentalmente de tres elementos: URL del Repetidor, dirección del *EndpointService* y *PeerGroupId*.

La sintaxis en código fuente es:

EndpointDestinationAddress = relayUrl + "/EndpointService:" + trimmedGID + "/" + PROXY_SERVICE_NAME + "/" + groupId;

El cambio se puede observar comparando esta dirección antes y después de unirse al grupo:

- *Antes de unirse al grupo:*

EndpointDestinationAddress:HTTP://127.0.0.1:9700/[1]EndpointService:JXTA-NetGroup/[2]urn:JXTA:uuid-DEADBEEFDEAFBABAFEEDBABE0000000E05/[3]urn:JXTA:JXTA-NetGroup[4]

- *Después de unirse al grupo:*

EndpointDestinationAddress:HTTP://127.0.0.1:9700/[1]EndpointService:uuid-E25359ECA0524ED28C2EF85FE16C004E02/[2]urn:JXTA:uuid-DEADBEEFDEAFBABAFEEDBABE0000000E05/[3]urn:JXTA:uuid-E25359ECA0524ED28C2EF85FE16C004E02[4]

1. La URL del Repetidor con el puerto del servicio (9700)
2. Indica el *groupId* pero con otra cabecera (*EndpointService*:)
3. Indica el nombre del servicio Proxy del relay
4. Indica el *peerGroupId*, que como se observa, cambia una vez el *peer* se une al nuevo *peerGroup.*

A pesar del aparente buen funcionamiento, se descubrió un fallo cuando se pretendía enviar un mensaje en este nuevo grupo y no había respuesta, lo cual causaba que el *peer* se quedara sondeando el repetidor indefinidamente. Por esta razón se examinó el código fuente del repetidor proporcionado por JXTA, específicamente el archivo (net\JXTA\impl\proxy\ProxyService.java). De acuerdo con el análisis realizado se concluyó que el servicio Proxy del relay no estaba siendo instanciado para los grupos diferentes del NetPeerGroup; por esta razón se agregaron las siguientes líneas al método handleJoinRequest de ProxyService.java, se recompiló la clase y se agregó a la librería correspondiente (/lib/JXTA.jar).

```
 private synchronized void handleJoinRequest(Requestor requestor,
                                              String grpId,String passwd) {
……..
try {
        // Fork this proxy into the new grp.
        p = new ProxyService();
        p.init(g, assignedID, implAdv);
        p.startApp(null);
       } catch(Exception e) {
        requestor.notifyError(e.getMessage());
        return;
        }..
... }
```

Finalmente, después de realizar el cambio, las pruebas efectuadas sobre las operaciones convencionales de creación, búsqueda y selección de tuberías, grupos y *peers* dentro del nuevo grupo, arrojaron resultados satisfactorios.

4.3 Diferencia entre las tuberías JXTAUnicast y JXTAPropagate definidas por JXTA

Teóricamente JXTA define tres tipos de tuberías:

- JXTAUniCast, las cuales son unidireccionales para comunicación punto a punto.
- JXTAPropagate, las cuales son unidireccionales para comunicación punto a multipunto.
- SecureUnicast, definidas como tuberías del tipo JXTAUnicast con la adición de una capa de seguridad SSL, usada para comunicaciones seguras.

Una prueba inicial consistió en crear una tubería de tipo JXTAUnicast, cuyo *pipeId* fuera conocido por todos los demás *peers*, de tal manera que todos estaban sondeando dicha tubería en espera de mensajes. De acuerdo con la definición, las tuberías JXTAUnicast están destinadas a las comunicaciones punto a punto, pero los resultados de las pruebas demostraron que al enviar un mensaje hacia la tubería creada, todos los *peers* que la estaban sondeando recibían el mensaje. Este fallo fue publicado en la lista de fallos (*bugs*) del proyecto JXTA y se discutió al respecto con varios desarrolladores de la comunidad JXTA en el canal #JXTA en el IRC. Se concluyó finalmente que las *pipes* JXTAUnicast y JXTAPropagate funcionan de la misma manera.

4.4 **Cancelación de la conexión de tuberías abiertas**

De acuerdo con la definición del método *close* de la clase *PeerNetwork*, su invocación cierra las tuberías abiertas en el proceso de lectura de datos desde un *peer* fuente. Sin embargo, las pruebas realizadas demuestran que esto no ocurre, debido a lo cual se generan inconvenientes que no permiten que un *peer* inicie sesión en un grupo diferente y deje de recibir mensajes provenientes de la tubería anterior, lo cual se traduce en tráfico innecesario que incrementa los costos de uso del servicio hacia el usuario final.

Al analizar el código fuente no fue posible dar una solución directa al problema, pero se propone una solución alternativa que busca eliminar el tráfico innecesario generado. En términos generales, al iniciar la aplicación P2P el usuario debe seleccionar el grupo al cual desea unirse después de haberse conectado a la red JXTA. Una vez ingresa al grupo, el *peer* crea dos *pipes*, una para recibir mensajes privados y otra para enviar y recibir mensajes públicos de todo el grupo. Para cambiar de grupo, el usuario debe reiniciar la aplicación. Teniendo en cuenta que las tuberías continúan existiendo por un cierto período de tiempo, por lo cual seguramente aún existan en un próximo ingreso, se propone que al iniciar la sesión en un grupo, el *peer* que acaba de ingresar envíe un mensaje multicast al grupo informando el identificador (*pipeId*) de la tubería privada y su nombre; de la misma forma, el resto de los *peers* deben responder enviando los mismos datos, de tal manera que todos los *peers* cuenten con información actualizada sobre los contactos en línea.

CONCLUSIONES

- El Punto de Encuentro Virtual surge como una alternativa a las aplicaciones de mensajería instantánea para dispositivos móviles existentes en la actualidad, que ofrece la robustez de un lenguaje como Java para extender sus posibilidades de servicio P2P hacia otros ámbitos relacionados como el comercio electrónico y el turismo, entre otros.

- La contribución más grande del proyecto JXTA y el subproyecto JXME a la comunidad en Internet es el desarrollo de un conjunto de protocolos para el soporte de aplicaciones P2P para dispositivos móviles que no sólo concentra la atención del desarrollador en la lógica del negocio de la aplicación, sino que estandariza los mecanismos de comunicación en un ambiente fuertemente heterogéneo como lo son las redes P2P.

- La naturaleza *Open Source* de JXTA/JXME hizo posible realizar algunas contribuciones a la comunidad de desarrolladores JXTA a nivel mundial relacionadas con la inclusión de los *peers* móviles a un grupo diferente del grupo por defecto, la conceptualización de la funcionalidad de los tipos de tuberías definidos en JXTA y la gestión de apertura y cierre de tuberías para habilitar la comunicación entre los diferentes *peers*.

- A través de las pruebas realizadas se puede inferir que las redes de 2.5G que actualmente operan en nuestro país ofrecen un ancho de banda suficiente para satisfacer los volúmenes de tráfico generados por una aplicación P2P de la naturaleza del Punto de Encuentro Virtual. Las verdaderas restricciones están asociadas al factor de mercado, en el cual los operadores de telefonía móvil deberán enfocar sus esfuerzos en los próximos años.

Referencias

[1] Brakeman, L. Peer-to-peer computing: a newly cool 'old' technology makes a comeback. *Heating/Piping/Air Conditioning Engineering: HPAC,* 2000, vol. *72*, p. 66.
[2] Campbell, A. Peer to Peer: Collaboration and Sharing over the Internet. *Technical Communication,* 2004, vol. *51*, p. 435.
[3] JXTA.org. JXTA *Project*. [Consulta: Junio de 2005]. Disponible en: HTTP://www.JXTA.org
[4] JXME.JXTA.org. JXME *Project*. [Consulta: Junio de 2005]. Disponible en: HTTP://JXME.JXTA.org.
[5] Muchow, J. *Core* J2ME *Technology &* MIDP. Palo Alto, California: Prentice Hall, p. 546.
[6] Androutsellis, S. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys,* 2004, vol. *36*, p. 335.
[7] Sing, L. (2002). *Making* P2P *interoperable: The* JXTA *story*. [Consulta: Junio 2005]. Disponible en: HTTP://www-128.ibm.com/developerworks/java/library/j-P2Pint1.html.
[8] Sing, L. (2002). *Making* P2P *interoperable: Creating* JXTA *systems*. [Consulta: Junio de 2005]. Disponible en: HTTP://www-128.ibm.com/developerworks/java/library/j-P2Pint3/, 2002.
[9] Sing, L. JXTA & peer-to-peer networks. *Dr. Dobb's Journal,* 2003, vol. *28*, p.30.
[10] Brendon, W. JXTA. Portland, New Riders, 2004, p. 488.
[11] Mattei, R. Getting the peers talking with JXTA. *Inside Solaris*, 2002, vol. *8*, p.1.
[12] Sun Microsystems (2005). JXTA *Programer Guide* v2.3. [Consulta: Junio 2005]. Disponible en: HTTP://www.JXTA.org/docs/JXTAProgGuide_v2.3.pdf.
[13] Salz, P. Calling P2P: Peer-to-Peer Networks Coming to a Phone Near You. *EContent*, 2005, vol. *28*, p.8.
[14] Faheem, K. (2004). *Wireless Messaging with* JXTA, *Part 1: Using* JXTA. [Consulta: Junio de 2005]. Disponible en: HTTP://www-106.ibm.com/developerworks/wireless/library/wi-JXTA/.

[15] ARORA, A., HAYWOOD, C. & SINGH, K. (2002). JXTA *for* J2ME – *Extending the Reach of Wireless with* JXTA *Technology*. [Consulta: Junio de 2005]. Disponible en: HTTP://www.JXTA.org/project/www/docs/JXTA4J2ME.pdf.
[16] SourceForge.net. *JM*IRC *Project*. [Consulta: Junio de 2005]. Disponible en: HTTP://jmIRC.sourceforge.net/
[17] SourceForge.net. *JIMM Project*. [Consulta: Junio de 2005]. HTTP://jimm.sourceforge.net/
[18] Jabber.org. *Jabber Foundation*. [Consulta: Junio de 2005]. Disponible en: HTTP://www.jabber.org/.
[19] Gryf.info. *Mobber Project*. [Consulta: Junio de 2005]. Disponible en: HTTP://gryf.info/mobber/en/