OPTIMIZACIÓN DE LAS PRESTACIONES DE LA PLATAFORMA DE COLABORACIÓN P2P: P2PEOPLE

Rafael Berenguer Vidal, Rafael Melendreras Ruiz, Ángel J. García Collado
Departamento de Ciencias Politécnicas – Grupo de Inv. en Telecomunicaciones Avanzadas
Universidad Católica San Antonio de Murcia
e-mail: [rberenguer,rmelendreras,ajgarcia]@pdi.ucam.edu

***Abstract*- In this paper, the performance optimization of the P2People platform is analyzed. For this purpose, the performance critical-points of P2P applications are examined firstly. Provided that P2People is based on P2P-JXTA architecture, the performance increase between JXTA releases is explored below. P2People optimization deals with the increase of its architecture throughput and the individual improvement of the different P2People modules. So, the performance increase strategies for these elements are shown and evaluated. Finally the experimental results probe that this optimization is well fitted.**

I. INTRODUCCIÓN

P2People [1] es una plataforma *open-source* basada en redes *peer-to-peer* (P2P) orientada a la formación de grupos de intereses comunes.

Un perfil *(profile)* definido por el usuario con sus intereses, es utilizado por el sistema para la localización del resto de los usuarios afines mediante un motor de búsqueda inteligente. Con los usuarios encontrados se van formando los grupos de intereses comunes de forma automática. La plataforma incluye asimismo un conjunto de servicios, basados en el paradigma P2P para permitir la comunicación y el trabajo colaborativo entre los usuarios de un mismo grupo.

Al contrario que la mayoría de las aplicaciones P2P [2], enfocadas a un solo servicio, P2People ofrece al usuario un conjunto de servicios colaborativos, que pueden ser ampliados por la comunidad *open-source*.

Esto conduce a una mayor complejidad en el sistema, formado por un gran número de componentes software que interaccionan entre ellos y a través de protocolos P2P con el resto de los usuarios de la red.

En el presente artículo se describen las estrategias que se han seguido con el fin de optimizar la plataforma para la reducción del tráfico de red, carga de procesador y memoria en los nodos, aspectos críticos de las aplicaciones P2P.

La estructura del artículo se establece como se indica a continuación: en la siguiente sección se describen los puntos críticos para las prestaciones de los sistemas P2P, en la sección III se describe cómo se ha optimizado cada uno de los módulos principales de la plataforma, en la sección IV se muestra los resultados de dichas optimizaciones, finalizando en la secciones V y VI con las futuras líneas de trabajo y conclusiones.

II. ANÁLISIS DE PUNTOS CRÍTICOS EN LAS PRESTACIONES DE SISTEMAS P2P

A. Aplicaciones peer-to-peer (P2P)

En una red P2P, cada nodo conectado a ella (*peer*) actúa como cliente y servidor al mismo tiempo, proporcionando acceso a parte de sus recursos (procesador, almacenamiento, etc.) y utilizando parte de los recursos de nodos vecinos [3].

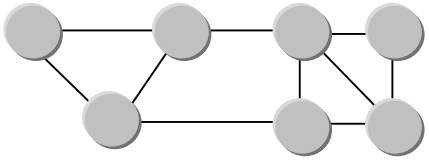

Fig. 1. Ejemplo de estructura de red P2P

Esta descentralización de la red consigue que la vulnerabilidad del sistema no se concentre en un único punto, mejorando su robustez frente a ataques. Asimismo, suele aparecer redundancia en la información al encontrase replicada en muchos *peers* de la red.

Con un diseño adecuado de los protocolos P2P, se puede garantizar la escalabilidad y balanceado de carga del sistema en su conjunto, proporcionando sistemas escalables altamente tolerantes a fallos.

En la actualidad existen numerosas implementaciones de sistemas P2P tanto en el ámbito de la investigación como comercial. Desde el punto de vista del rendimiento, su elemento crítico reside en que los nodos actúan como servidor (proporcionando servicios al resto de usuarios) y cliente (haciendo uso de los servicios del resto de usuarios).

Asimismo, cada *peer* de la red, ha de ser capaz de encontrar qué usuarios pueden proporcionarle el servicio solicitado. Si, además, este concepto se aplica para redes P2P puras (sin ningún servidor o *super-peer* que actúa como índice de información), cada nodo deberá atender las consultas del resto de los nodos. Y dado que es frecuente que no exista conectividad total entre todos los nodos, cada uno deberá actuar de intermediario entre nodos contiguos retransmitiendo las solicitudes de servicios.

Otra tarea importante a realizar por cada nodo, es el mantenimiento de las conexiones con el resto de los usuarios. Puesto que la entrada y salida de la red es aleatoria, cada nodo suele identificarse por un identificador lógico (sobrenombre o ID generado aleatoriamente) en lugar de una dirección IP. Así, los nodos tendrán que ser capaces de realizar este mapeo entre identificadores de forma dinámica y transparente al equipamiento de red (*routers* y *firewalls*).

Los nodos deberán proporcionar una calidad de servicio (QoS) en el suministro de los servicios. Si no pueden atender adecuadamente todas las peticiones, deberán crear una cola donde situarlas y aplicar una política para su gestión.

Además de estas tareas de búsqueda de servicios, gestión y manteniendo de las conexiones y QoS, cada nodo deberá proveer el servicio a los nodos que lo soliciten.

Es importante destacar que el número de usuarios en cada plataforma P2P tiende a ser elevado. Como además los nodos suelen ser máquinas de prestaciones estándar en las que la aplicación P2P suele ejecutarse en un segundo plano, el rendimiento y consumo de recursos por parte de la aplicación es un aspecto muy crítico para su funcionamiento.

Por tanto, se deberá optimizar el rendimiento de las aplicaciones, minimizando el consumo de recursos (en concreto carga de procesador y memoria, y tráfico de red) de las tareas de gestión, para que estos recursos puedan ser utilizados principalmente para la provisión de los servicios.

B. Implementación Java P2P (JXTA)

JXTA es una plataforma de interconexión y desarrollo P2P desarrollada por Sun y basada en código abierto. Se ha convertido en un protocolo estándar en comunicaciones P2P [4], existiendo ya un variado número de aplicaciones que lo incorporan. Su optimización es fundamental para todas las aplicaciones que hacen uso de dicha plataforma como subcapa de red P2P.

La versión 1 de JXTA junto a los proyectos que se han desarrollado sobre ella han permitido una gran optimización de su rendimiento reflejada en la versión 2 actual [5].

JXTA 1 adopta una aproximación teórica a la topología de red física, lo que provoca inundación de mensajes. En cambio JXTA 2 se adapta a una estructura de red jerárquica. Así, se define el concepto de *rendezvous super-peer network* [6] que segrega la red en subredes P2P, permitiendo el paso de mensajes sólo cuando sea realmente necesario. De esta forma se aumenta la escalabilidad del sistema.

Mientras que en JXTA 1 los servicios ofertados son publicados periódicamente mediante mensajes de *advertisements* al resto de usuarios [4], JXTA 2 utiliza el algoritmo *shared resource distributed index* (SRDI) para crear y mantener un índice conceptual de servicios en la red de forma distribuida. El SRDI gestiona una tabla hash con servicios y sus proveedores que se mantiene de forma distribuida entre los nodos, reduciendo el número de *advertisements* enviados en la solicitud de servicios en la red [5].

En JXTA 2, se han rediseñado los módulos críticos para el rendimiento de la plataforma: aumento de eficiencia en la gestión de los *threads*, eliminación de las múltiples copias de buffer en pila de protocolos, control sobre los *advertisements* almacenados en la caché local, almacenamiento mediante base de datos *Xindice btree*, etc.

Todas estas mejoras permiten que las aplicaciones que se construyen sobre JXTA reduzcan la cantidad de recursos destinados a la gestión de los servicios, aumentando sus prestaciones en la provisión de los mismos.

C. Plataforma P2People

P2People está construido sobre JXTA 2 y permite a los usuarios formar grupos de intereses comunes [1] y la comunicación y el trabajo entre los usuarios de un mismo grupo mediante servicios colaborativos basados en el paradigma P2P. P2People hace uso de los *peer-groups* de JXTA, consistentes en agrupaciones lógicas de usuarios con una colección común de servicios. Éstos son utilizados para la formación de grupos de "intereses comunes" y para proporcionar los servicios colaborativos a usuarios del grupo.

El funcionamiento de P2People se descompone en los siguientes pasos: creación del perfil (*profile*) con los intereses del usuario, búsqueda de usuarios mediante el algoritmo de *matching* y uso de herramientas colaborativas.

La optimización de P2People pasa en primer lugar por un aumento del rendimiento de su capa P2P descrita en la sección II-B. Asimismo, es necesario un correcto diseño de la arquitectura que simplifique la interrelación entre los distintos servicios. Por último, será necesario optimizar los distintos elementos, y en concreto aquellos que mayor consumo de recursos requieran. Estos conceptos son descritos en la siguiente sección.

III. ESTRATEGIAS DE OPTIMIZACIÓN DE P2PEOPLE

A. Arquitectura

La arquitectura está formada por: núcleo, servicios, interfaz gráfico (GUI) y la pila de protocolos P2P (JXTA).

El núcleo tiene como tarea fundamental la gestión de toda la plataforma, teniendo acceso a todos los servicios y permitiendo la comunicación entre ellos. El resto de las tareas están gestionadas por uno o varios servicios.

Éstos se dividen en funcionales y colaborativos. Los primeros realizan las tareas principales de la aplicación: gestión y creación de los grupos, mantenimiento de conexiones entre usuarios y otros aspectos relacionados con la seguridad y la reputación. Los colaborativos se utilizan para que los usuarios se comuniquen y colaboren entre ellos.

Con el fin de aumentar las prestaciones, esta arquitectura ha evolucionado desde un modelo inicial clásico basado en capas horizontales funcionales *(Horizontal Layers* - HL*)* con los servicios colaborativos situados por encima (Fig. 2), hasta un diseño vertical *(Vertical Components* - VC*)* basado exclusivamente en servicios JXTA -tanto funcionales como colaborativos-, situados directamente sobre JXTA y gobernados por una matriz de interconexión (Fig. 3).

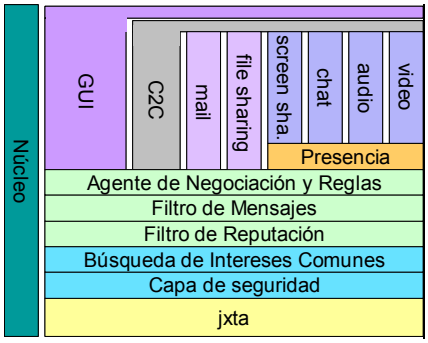

Fig. 2. Aproximación HL para la arquitectura de la plataforma P2People.

Con el modelo HL los componentes de colaboración tienen que atravesar obligatoriamente todas las capas funcionales provocando retardos en cualquier proceso de comunicación entre servicios. Asimismo, dado que las dependencias entre los distintos objetos son rígidas, la carga de procesador y consumo de memoria en los nodos es elevado.

Por contra, con el modelo VC todos los servicios acceden directamente a JXTA para la comunicación con resto de usuarios. El número de capas a atravesar es mínimo, con lo que los retardos son considerablemente inferiores.

Por otra parte, las relaciones entre servicios vienen gestionadas por la matriz de interconexión de servicios *(services matrix component)* [1]. El traspaso de información entre los servicios se realiza mediante el intercambio de mensajes, solamente entre los componentes que lo necesiten.

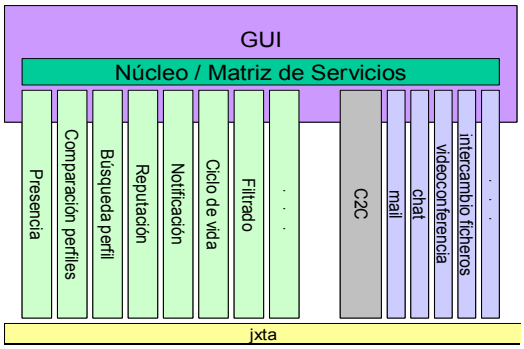

Fig. 3. Aproximación VC para la arquitectura de la plataforma P2People.

B. Servicio de presencia

La principal tarea del servicio de presencia consiste en indicar en qué estado se encuentran los usuarios del sistema: conectado, desconectado, ocupado, etc. Dado que es utilizado por el resto de los servicios, para permitir o no su activación, a este servicio se le exige una elevada fiabilidad.

El funcionamiento básico de este servicio consiste en el envío periódico de un *advertisement* indicando el estado del usuario al resto de *peers*. Cualquier nodo al recibir este mensaje, muestra su estado en el GUI e inicia un timer t_p. Si vence el timer (tiempo de expiración t_e) sin recibir otro mensaje, entiende que el usuario está desconectado.

Dado que una elevada frecuencia de refresco f_r se contrapone con el ancho de banda requerido para el envío de mensajes, es necesario llegar a una solución de compromiso.

En una primera aproximación se ha fijado un t_e y f_r fijos $(t_e = 2 / f_r)$. No obstante, con el fin de conseguir adaptación a las condiciones cambiantes de la red y reducción del tráfico, en una segunda aproximación se ha optado por utilizar un esquema basado el algoritmo de Karn [7] que tiene en cuenta el RTT de la red. Adicionalmente, se han incluido otras optimizaciones basadas en los requerimientos del servicio.

Con esta segunda estrategia, el rendimiento de la aplicación aumenta al tener una mayor f_r cuando las condiciones cambiantes de la red o usuarios así lo requieren, reduciendo el consumo de recursos cuando no son necesarios.

C. Algoritmo de matching (MA)

Todo *peer* ejecuta el servicio de comparación de perfiles aplicando el algoritmo de *matching*. Éste proporciona como resultado un índice de correspondencia entre perfiles.

El perfil se ha definido como un árbol de palabras clave *(key words)*. Esta organización de la información de forma jerárquica, está enfocada a la rapidez del algoritmo. Al tener la información con esta estructura, encontrar perfiles compatibles se reduce a realizar un *string matching* entre palabras clave de los árboles.

En una primera aproximación, el funcionamiento del algoritmo consiste en comparar los árboles de forma completa. No obstante, dado que para ejecutar el servicio de *matching* los *peers* deben encontrarse dentro del mismo grupo JXTA, con un gran número de usuarios en el sistema, puede surgir un problema de escalabilidad [4].

Para reducir el tamaño de los grupos, se ha jerarquizado el árbol en dos niveles En primer lugar, tendremos un árbol de áreas temáticas fijas y dentro de cada área, el usuario podrá definir un subárbol de intereses asociados a dicha área. Cada área temática se asocia a un grupo JXTA y sobre los usuarios que participan en una determinada área se aplica el algoritmo de *matching*. Con este planteamiento, el MA se aplicará de forma independiente en cada área temática (Fig. 4) reduciéndose el número de *peers* por grupo JXTA.

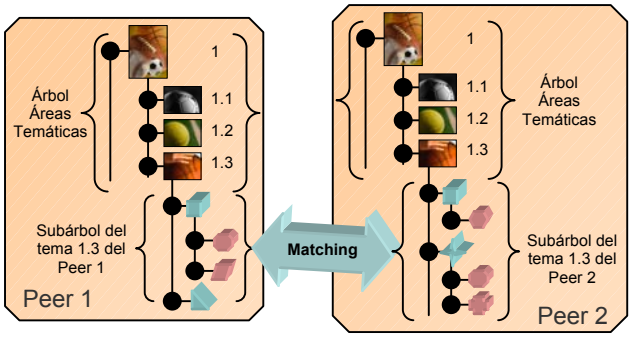

Fig. 4. Algoritmo de *matching* con árbol de áreas temáticas y subárboles.

En [1] se describe el algoritmo de *matching* entre los subárboles. En este algoritmo se ha buscado potenciar la velocidad sobre la precisión del resultado.

Se asigna un peso a cada profundidad de los nodos de los subárboles y se realiza un *string matching* entre palabras clave entre diferentes nodos. El índice global de *matching (gmi)* que se obtiene, es función de la semejanza entre palabras clave $s_{i,j}$, sus pesos w_i, w_j y compatibilidades entre nodos superiores e inferiores $s_{i',j'}$:

$$gmi_{a,b} = \frac{1}{N} \sum_i \sum_j f(s_{i,j}, w_i - w_j, s_{i',j'}) \qquad (1)$$

Para implementar el *string matching* entre las palabras clave, se ha optado por el algoritmo *Ratcliff/Obershelp pattern recognition* [8] por su adecuado compromiso entre velocidad y resultados.

D. Servicio de videoconferencia

La videoconferencia es uno de los servicios colaborativos de P2People con mayor requerimiento de recursos ya que suministra contenido multimedia en tiempo real. Así, su optimización es fundamental para un uso eficiente.

Este servicio está basado en las librerías *Java Media Framework* (JMF) de Sun [9]. Incorpora diversos códecs de audio y vídeo, y permite captura, distribución y reproducción en tiempo real mediante protocolo RTP.

El punto crítico del servicio en P2People es la transmisión de los datos. Todas las comunicaciones entre servicios JXTA se realizan mediante *pipes* JXTA [4]. Éstos permiten que las direcciones IP de los nodos sean transparentes a los servicios ya que es JXTA quien gestiona el mapeo entre direcciones IP e identificadores de usuarios, permitiendo la transmisión de datos entre nodos situados en redes IP distintas.

Sin embargo, al establecer la transmisión sobre los *pipes* se introducen unas elevadas latencias. Esto es debido a que no se pueden aprovechar las características del RTP.

La alternativa a esta aproximación es la difusión de los

datos directamente sobre sockets IP/UDP mediante filosofía cliente/servidor. Así, los rendimientos de transmisión son excelentes, pero por el contrario no hay posibilidad de comunicación entre nodos con IPs locales en redes distintas.

La solución propuesta en [10] es el empleo conjunto de ambas estrategias, utilizando el envío sobre sockets estándar cuando las características de la red lo permitan y los *pipes* JXTA cuando no sea posible establecer conectividad directa.

IV. ANÁLISIS DE PRESTACIONES Y RENDIMIENTO

Para analizar el rendimiento de las dos aproximaciones para la arquitectura, se han realizado mediciones de un conjunto de variables (carga de procesador, memoria, ancho de banda) al ejecutar diferentes servicios de la plataforma. Estas medidas han sido realizadas sobre grupos de 46 máquinas con ejecución simultánea de la aplicación.

En la figura 5 se representa el valor medio e intervalo de variación de los tiempos de respuesta en la inicialización de las áreas temáticas, ejecución del algoritmo de *matching* tras la modificación de los perfiles y notificación al resto de usuarios de cambio en el estado de presencia. Con la arquitectura HL dichos valores son mayores que con la aproximación VC. Por este motivo, se ha optado por el uso de esta última arquitectura para la plataforma.

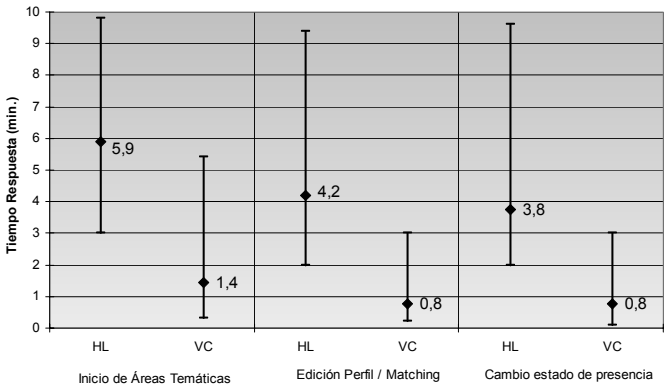

Fig. 5. Comparativa de tiempos de respuesta medidos para tres acciones con arquitectura de capas horizontales (HL) y componentes verticales (VC).

De igual forma, con el fin de analizar la escalabilidad de la segunda aproximación para el algoritmo de *matching* propuesto, se han medido las variables anteriormente descritas ejecutando el proceso de *matching* variando el número de usuarios conectados al área temática.

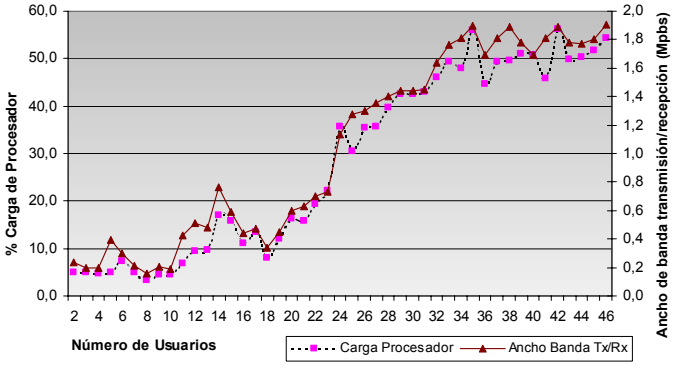

Fig. 6. Valor medio de carga de procesador y ancho de banda consumido en cada nodo en función del número de usuarios en la misma área temática.

En la figura 6 vemos que aunque la carga de procesador y ancho de banda crecen a medida que aumenta el número de *peers* en el área temática, se llega a unos valores máximos que se mantienen constantes aunque el número de *peers* siga aumentando con lo que se garantiza la escalabilidad.

V. LÍNEAS DE TRABAJO FUTURAS

En la actualidad se está investigando en la mejora del algoritmo de *matching*, con el fin de dotarle de una mayor inteligencia que permita definir los intereses en lenguaje natural sin un aumento de los requerimientos en los nodos.

Asimismo se trabaja en la optimización de los restantes módulos de la plataforma -no analizados a fondo hasta el momento- como son los servicios de búsqueda *(queries)*, procesamiento y cálculo de la reputación distribuida, optimización del ciclo de vida de los *advertisements,* así como optimizaciones software de componentes colaborativos como compartición de escritorio a través de mejora del códec de compresión, intercambio de ficheros para soportar *multithreating* para descargas múltiples, etc.

VI. CONCLUSIONES

En este artículo se han mostrado los puntos críticos desde el punto de vista de las prestaciones que presentan las aplicaciones P2P y en concreto la plataforma P2People desarrollada sobre la arquitectura JXTA. Se ha analizado la influencia que éstos pueden tener sobre el rendimiento de las aplicaciones en redes P2P con un gran número de usuarios.

En segundo lugar, se ha estudiado la plataforma P2People, analizando sus módulos críticos para el rendimiento de la aplicación y las estrategias utilizadas en su optimización. Finalmente, se han mostrado los resultados que demuestran que estas estrategias han conducido a una mejora de su rendimiento.

AGRADECIMIENTOS

Este trabajo ha sido financiado por la Unión Europea dentro del programa IST a través de P2People-2002-38458.

REFERENCIAS

[1] R. Melendreras Ruiz, R. Berenguer Vidal, A.J. García Collado. "Desarrollo de la plataforma de colaboración basada en grupos de «intereses comunes» y redes peer-to-peer: P2People", *Revista IEEE América Latina*, vol. 2 issue: 1, Mar. 2004.

[2] C. Shirky, K. Truelove, R. Dornfest, L. Gonze, "P2P networking overview: The emergent P2P platform of presence, identity and edge resources", O'Reilly, 2001

[3] L. Gong, "Peer-to-peer networks in action", *IEEE Internet Computing*, vol. 6 issue 1, pp. 37–39, Jan/Feb. 2002

[4] L. Gong, "JXTA: A network programming environment", *IEEE Internet Computing*, vol. 5(3), pp. 88-95, May/Jun. 2001

[5] S. Li, "JXTA 2: A high-performance, massively scalable P2P network", Wrox Press, www.ibm.com, 11 Nov. 2003

[6] B. Traversat, A. Arora, M. Abdelaziz, M. Duigou, C. Haywood, J-C. Hugly, E. Pouyoul, B. Yeager, "Project JXTA 2.0 Super-Peer Virtual Network", www.sun.com, 25 May. 2003

[7] P. Karn, C. Partridge, "Improving Round-Trip Estimates in Reliable Transport Protocols", *ACM Transactions on Computer Systems,* Nov. 1991

[8] J.W. Ratcliff, D. Metzener, "Pattern Matching: The Gestalt Approach", Dr. Dobb's Journal, pp. 46, Jul. 1988

[9] Java Media Framework API Guide, http://java.sun.com

[10] M. Meza, B. Marusic, S. Dobravec, J. Tasic. "Peer to Peer Search Engine and Collaboration Platform Based on JXTA Protocol", *Proc. 2003 IEEE Eurocon International Conference on Computer as a Tool*, Ljubljana, Slovenia