SOAD: AN APPLICATION FOR PEER TO PEER WEB COMMUNICATION BETWEEN USERS IN A NETWORK

Muturi, I.M.
Chuka University, P. O. Box 109-60400, Chuka
Email: mwauraisaac9@gmail.com; Tel.: 0715934415

Citation: Muturi, I.M. (2016 Soad: An application for peer to peer web communication between users in a network. In: Isutsa, D.K. and Githae, E.W. *Proceedings of the Second Chuka University International Research Conference held in Chuka University, Chuka, Kenya from 28th to 30th October, 2015. 338-344pp.*

ABSTRACT
Web technology has emerged as effective tool for communication, socialization, e-commerce, e-government, research, mass awareness and information sharing. With advancement in technology such as Wi-Fi many people access this technology. Today, many places have Wi-Fi network, but people haven't fully exploited it due to cost and limited connectivity. It is possible to scan a Wi-Fi network and discover no connection on it. There is thus a need for developing a system that allows free and secure communication between users without interfering with current networking protocols and allowing for interoperability, interactivity and usability between users. The paper presents the SoAd application as a proof of concept in implementing such a system. SoAd was developed using java which is platform independent and supported by many operating systems including mobiles such as Android™. Java.net class is a rich networking tool for socket and URL communication in local and internet networks. SoAd emulates a server as a client and allows communication between users in the network using java sockets. The idea is to allow users communicate without a centralized server when in a network and delivery of information as web contents. This application suits e-commerce and socialization among many others.
Keywords: Java.Net, Web, Local Network, Socket Communication, e-Commerce

INTRODUCTION
SOAD is a compound word of social and ad hoc. An ad hoc network refers to a network connection established for a single session and does not require a router or a wireless base station. Ad hoc networks are used for a specific purposes, such as sharing documents during a meeting or playing multiplayer computer games. SOAD application is developed simulating an ad hoc network for communication and sharing information between the peers. With advancement in technology more people have accessed telecommunication devices either as personal computers, personal digital assistance (PDAs), mobile phones, beepers, sensors, switches, wearable computers, telemetry sensors, or tracking agents. There is thus

a great demand for engineering a peer to peer communication system between these devices. An interoperable, platform independent, ubiquity, secure and monitored system. The base idea is to achieve a system that eliminates the need of having a centralized server between users and allowing all devices regardless of their platform to communicate freely when they are linked in a network thus creating a virtual society. This research developed the SOAD application which integrates a web application as the graphical user interface and JXTA™ technology as a protocol to achieve peer to peer communication.

Related work

Ad hoc and peer to peer communication has widely been used for sharing of files, resources sharing and communication. Example Napser and Bluetooth. The concept of peer-to-peer computing was envisioned in earlier software systems and networking discussions, reaching back to principles stated in the first Request for Comments, RFC. Also in early development of the World Wide Web, the World Wide Web was close to a P2P network in that it assumed each user of the web would be an active editor and contributor, creating and linking content to form an interlinked "web" of links. Peer to peer communication mostly is built on top of TCP/IP protocol, where users communicate using TCP/IP ports and respective IP addresses of their peers to communicate. In other systems and application such as skype™ other proprietary protocols are employed that enable discovery and communication of peers.

JXTA™ Technology

JXTA is an open source project for open network computing platform designed for peer-to-peer (P2P) computing by way of providing the basic building blocks and services required to enable anything anywhere application connectivity. It creates a virtual network overlaying on top the existing physical network infrastructure and allow messages exchange with any other peers independently of its network location (firewalls, NATs or non-IP networks). Messages are transparently routed, potentially traversing firewalls or NATs, and using different transport/transfer protocols (TCP/IP, HTTP) to reach the receiving peers. Thus JXTA protocols are transport protocols and programming languages independent. They can be implemented in the Java, C/C++, .NET, Ruby, and numerous other languages on top of TCP/IP, HTTP, Bluetooth, and other network transports all the while maintaining global interoperability.

JXTA protocols provide a standard way in which peers:

- Discover each other dynamically across firewalls and NATs.
- Self-organize into peer groups and provide a service
- Advertise and discover network resources in a network.
- Securely communicate with each other.
- Monitor peer activities remotely.
    - Find available content at network sites.

JXTA Network

JXTA network is composed of:

- Peers- a node or any type of device connected to a network that implements any JXTA protocol.
- Peer group-a collection of peers that have agreed upon a common set of services, or interests.
- Pipes- an asynchronous, unidirectional and non-reliable virtual communication channels used as transfer mechanism for of objects.
- Messages- the basic unit of exchange between peers that JXTA services and applications use to communicate.
- Advertisements-XML documents used to advertise network resources (peer, pipe, data, peer group, etc.)
- JXTASockets- are bidirectional (full duplex) pipes which implements the java.net.Socket interface used to send and receive data as stream.
- JXTA IDs- a uniform peer addressing scheme and location independent logical addressing model. All network resources are assigned a unique JXTA ID.

JXTA uses an abstraction of pipes to peers, and peers to endpoints, without reliance upon a central naming/addressing authority such as DNS. It has a decentralized search infrastructure based on Distributed Hash Table (DHT) for resource indexing. Messages are transparently routed, potentially traversing firewalls or NATs, and using different transport/transfer protocols (TCP/IP, HTTP) to reach the receiving peers

Peers discovery and peer groups

SOAD application discovers peers using JXTA™ peer discovery protocol. The application sends peer discovery advertisements to any peers in the network as well as receive advertisements from other peers. The received advertisement is read and the ID of the sender taken. The application checks whether the sender is a member of its peer group and opens a JXTA socket towards the peer for communication. Each application is a member of world peer group which acts as a pool for discovering other new peers in range/network. To achieve security each peer has its own peer group which is managed and monitored by the peer. Thus any other peer wishing to communicate to the peer must be a peer member of the peer's peer group hence must have been accepted by the peer. The application thus has to maintain the peer IDs of its clients in a database. See fig. 3.2

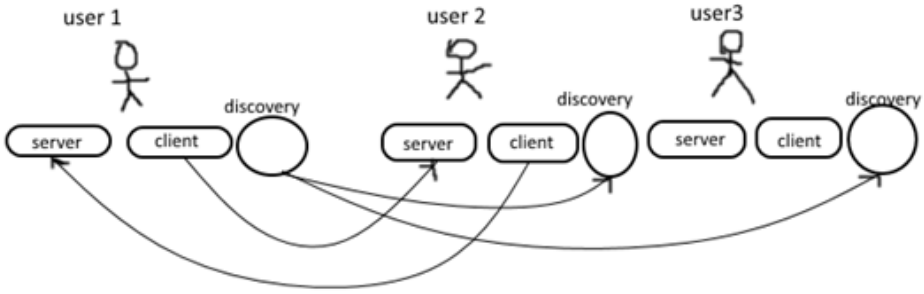


Figure 3.2: Peer sending discovery messages.

JXTA Sockets and Http

In this application, JXTA sockets are used as Java socket in sending and receiving HTTP requests. JXTA sockets differ from Java sockets since they must flush data at the end of data transmission, send data using peer ID and advertisements rather than TCP sockets and doesn't implement keep alive. There are two models of the socket used:

- JXTA client socket model for sending HTTP requests. It binds to a server socket using the server socket ID and the server's advertisement.
- JXTA server socket model used processing HTTP requests and sending response to the requesting client that binds to it.

At the base of the client socket is HTTP client and a HTTP server at the base of JXTA server socket. See Figure 2.1

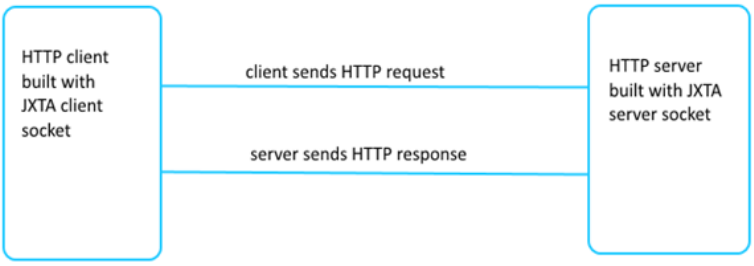


Figure 2.1: Sending and receiving HTPP request and responses

JXTA proxy server

There is need of developing a mini proxy server since the user interacts with the application using normal TCP socket from a browser and also needs to interact with other JXTA socket servers. The mini proxy server has a local web server -Tomcat Apache™ that is embedded thus making SOAD a standalone web application. It is used in serving of JSP pages, storage of databases, hosting servlets and serving other web contents such as java scripts, CSS, and images.

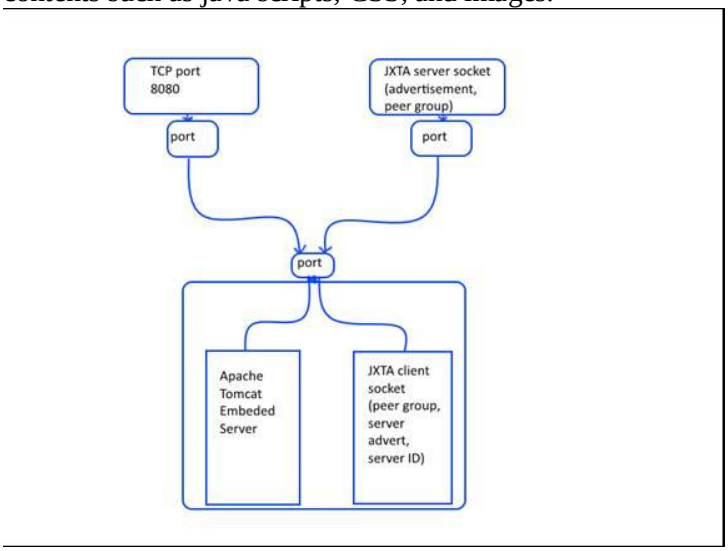


Figure 2.2: Proxy Server

Figure 2.2 shows a representation of the proxy server. It is clear from the figure that only the local client (browser(s)) shall access the web server through the main proxy server TCP port 8080. Other peers will connect to the proxy server by binding with the JXTA server socket using the socket advertisement and who belongs to the peer group of the hosting peer.
All the request are served using multi-threaded ports from the proxy server. Each thread makes a request to either the embedded web server or the JXTA client socket after the proxy server reads (parse) the HTTP request and determines the name of the host. If the host name of the request corresponds to the peer ID of the hosting peer, it connects the thread to the local web server (embedded apache) else, the thread is connected to a JXTA client socket to another peer's server socket.

THE WEB APPLICATION

SOAD uses a web application to represent the activities undertaken by JXTA such as peer discovery as well as providing an interactive user interface for the user. A web app suits as a user interface since it is cross platform, require less skill to use, provide interoperability and an interactive beautiful user interface. The web app is a single webpage application that auto-generates dynamic contents according to the users interaction (reactive). This is possible by use of Java servlets, JSP (Java server Page™), Java beans, and other web design tools like JavaScript and CSS. All the web contents are hosted in an embedded Apache™ Tomcat server.

JAVA SERVER PAGES™ (JSP) TECHNOLOGY

Java Server Pages™ (JSP) technology allows mixing of HTML and Java codes in a web page. A JSP pages contains HTML codes, blocks of Java code (scriplets) and other expressions. These JSP files are translated by JSP interpreters (jasper) into Java source code. The resulting source code is then compiled, resulting in class files that can be loaded and executed by a Java virtual machine and generate HTML file. See Figure 3.1

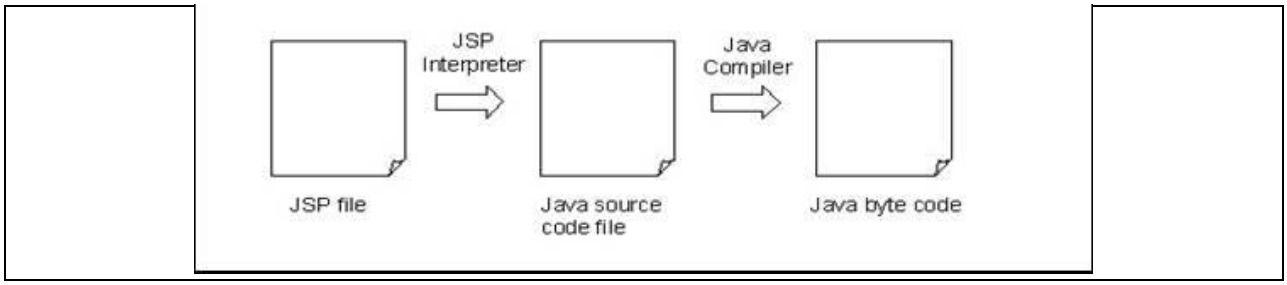
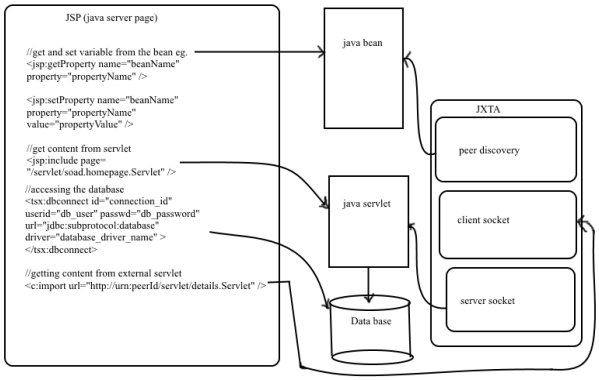


Figure 3.1: Compilation of JSP page

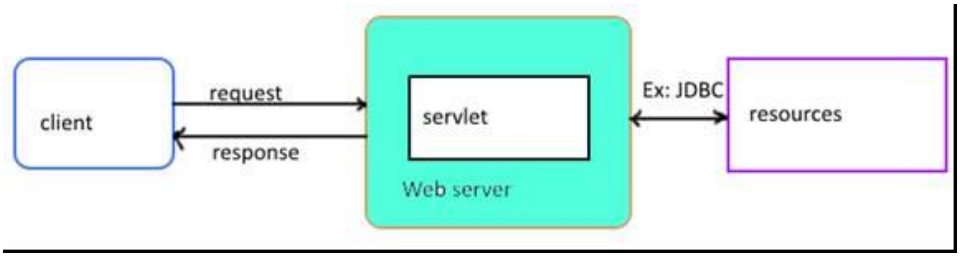
JSP used in this application to make the web application dynamic by getting data from Java beans, getting page divisions from the Java servlets, setting and managing sessions, reading data from data base and generating HTLM codes for the webpage. To make the web app a single webpage application the content supplied from external servers have must not replace the existing webpage. When a user clicks on a link on the webpage, the web contents received is displayed on a division where the link pre-existed as a new division. Thus the user interacts with other web pages from his/her web page.



The JSP page in this application is a backbone for requesting content from the servlet, the local servlets or peer servlets, getting data from the beans and other web tools such as CSS, JavaScript and images.

JAVA SERVLET

Java servlets are multithreaded server-side applications used for serving dynamic content for incoming HTTP requests, session management and integrating J2EE architecture. Java servlet is a class implementing the javax.servlet.Servletinterface or the javax.servlet.http.HttpServlet class -an abstract class that provides common abstractions necessary for all servlets communicating with clients via HTTP. When a client sends a request to a servlet-enabled server that invokes a servlet, the server checks to see if the servlet is loaded. This application uses java servlet to serve content to HTTP request between peers and in session management for security. This application uses servlets to supply dynamic web content and in session management. All requests from peer are served with a servlet. The servlet sends a webpage division to peer so as not to tamper with webpage of the requesting peer.



JAVA BEANS

JavaBeans are used as container for the dynamic data and objects. In this application JavaBeans are used as an interface between JXTA, database and web app. After instantiation of Java beans, JXTA will add or remove data such the names of peer and their IDs (urns) available during a session which will is made available as content to the JSP page (properties). In JSP properties can be read and set through the bean's accessor methods -Java methods named according to the JavaBeans conventions. The bean encapsulates all information about the item it represents in one simple package.

DISCUSSION

SOAD application allows for free discovery of peers and transfer of information between users linked in a network. The web application is user friendly and gives abstraction of the activities undertaken by JXTA. This application can be used many areas such as:

(i) E-commerce. Mobile telecommunication devices such as mobiles phones are the most suitable target for E-commerce. SOAD captures these devices by allowing them communicate using JXTA and hence. Business and firms can use this app to automate their processes and advertise themselves.

(ii) File sharing. User share files such as images by uploading the file to the peers site.

(iii) Socialization. SOAD create a virtual society to all users in the network. Users can freely communicate to each other.

(iv) Reducing the digital split. SOAD provide free communication to all users provided they are linked to a network. This will attract people to information technology hence reducing the digital split.

(v) Information sharing. Due to free communication, users will be encouraged share ideas and information between each other.

(vi) Multi-media. Multi-media virtual peer can set up in public networks that allow access of multi-media contents such as digital newspapers.

CONCLUSION

In this paper I have described how to successfully establish a peer to peer web communication between peers who are linked to a network. I have described how to discover other peers in the network using JXTA protocols. If this system is to be implemented, the benefits of the currently used World Wide Web may be realized but after period of time since the sites/contents highly depends on presence of users. The more the users the more the content. It is also important to note that, this system rely on a network link. A link, such as Bluetooth, Wi-Fi, or Ethernet must exist and for a peer to communicate to another, both must be in a network. In this project, SOAD only requests for a webpage only after it has discovered that the peer is on or available in the network. This eliminates the probability of requesting content from an unavailable peer. Other network properties such as QoS (quality of service) and speed of connection shall all depend on the nature of network the peer is linked on.

REFERENCES:

Bogdan, C., Gabriel-Miro, M. 2003. Advanced Network Programming –Principles and Techniques Network Application Programming with Java

Computer Communications and Networks ISBN 978-1-4471-5291-0 ISBN 978-1-4471-5292-7 (eBook) DOI 10.1007/978-1-4471-5292-7 Springer London Heidelberg New York Dordrecht

David, R. and Michael, R. 2002. Java™ Network Programming and Distributed Computing. Addison Wesley

Hans Bergsten (August 2002) JavaServer Pages™, 2nd Edition. O'Reilly

http://platform.jxta.org/java/currentwork.html

http://jxta.dev.java.net

http://ibm.com/redbooks

JXTA (September 10th, 2007) Java™ Standard Edition v2.5 Programmers Guidehttps://jxta-docs.dev.java.net/jxse-javadoc/current/jxse/api/index.html

"Peer to peer network", https://en.wikipedia.org/wiki/Peer-to-peer

Rüdiger Schollmeier. 2002. A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications, Proceedings of the First International Conference on Peer-to-Peer Computing, IEEE.

Liang, S. and Bracha, G. 1998. Dynamic Class Loading in the Java(tm) Virtual Machine", in ACM OOPSLA'98

Subrahmanyam A. Ph.D.(n.d) Java Server Programming: Principles and Technologies

Ueli Wahli, Mitch Fielding, Gareth Mackown, Deborah Shaddon, Gert Hekkenberg