ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

A Mobile Peer To Peer Architecture Services in Social Network Applications

Kanaparthi Kedari
Nimra College of Engineering & Technology.

Rehana Begum
Nimra College of Engineering & Technology

ABSTRACT:

Social network applications are becoming increasingly popular on mobile devices. A mobile presence service is an essential component of a social network application because it maintains each mobile user's presence information, such as the current status (online/offline), GPS location and network address, and also updates the user's online friends with the information continually.

If presence updates occur frequently, the enormous number of messages distributed by presence servers may lead to a scalability problem in a large-scale mobile presence service.

To address the problem, we propose an efficient and scalable server architecture, called Presence Cloud, which enables mobile presence services to support large-scale social network applications. When a mobile user joins a network, Presence Cloud searches for the presence of his/her friends and notifies them of his/her arrival.

Presence Cloud organizes presence servers into a quorum-based server-to-server architecture for efficient presence searching. It also leverages a directed search algorithm and a one-hop caching strategy to achieve small constant search latency. We analyze the performance of Presence Cloud in terms of the search cost and search satisfaction level.

The search cost is defined as the total number of messages generated by the presence server when a user arrives; and search satisfaction level is defined as the time it takes to search for the arriving user's friend list. The results of simulations demonstrate that Presence Cloud achieves performance gains in the search cost without compromising search satisfaction.

1 INTRODUCTION:

Because of the ubiquity of the Internet, mobile devices and cloud computing environments can provide presence-enabled applications, i.e., social network applications/services, worldwide. Facebook Twitter Foursquare Google Latitude buddycloud and Mobile Instant Messaging (MIM) are examples of presence-enabled applications that have grown rapidly in the last decade. Social network services are changing the ways in which participants engage with their friends on the Internet.

They exploit the information about the status of participants including their appearances and activities to interact with their friends. Moreover, because of the wide availability of mobile devices (e.g., Smartphones) that utilize wireless mobile network technologies, social network services enable participants to share live experiences instantly across great distances. For example, Facebook receives more than 25 billion shared items every month and Twitter receives more than 55 million tweets each day. In the future, mobile devices will become more powerful, sensing and media capture devices. Hence, we believe it is inevitable that social network services will be the next generation of mobile Internet applications.

A mobile presence service is an essential component of social network services in cloud computing environments. The key function of a mobile presence service is to maintain an up-to-date list of presence information of all mobile users. The presence information includes details about a mobile user's location, availability, activity, device capability, and preferences. The service must also bind the user's ID to his/her current presence information, as well as retrieve and subscribe to changes in the presence information of the user's friends. In social network services, each mobile user has a friend list, typically called a buddy list, which contains the contact information of other users that he/she wants to communicate with.

The mobile user's status is broadcast automatically to each person on the buddy list whenever he/she transits from one status to the other. For example, when a mobile user logs into a social network application, such as an IM system, through his/her mobile device, the mobile presence service searches for and notifies everyone on the user's buddy list. To maximize a mobile presence service's search speed and minimize the notification time, most presence services use server cluster technology. Currently, more than 500 million people use social network services on the Internet. Given the growth of social network applications and mobile network capacity, it is expected that the number of mobile presence service users will increase substantially in the near future. Thus, a scalable mobile presence service is deemed essential for future Internet applications.

In the last decade, many Internet services have been deployed in distributed paradigms as well as cloud computing applications. For example, the services developed by Google and Face book are spread among as many distributed servers as possible to support the huge number of users worldwide. Thus, we explore the relationship between distributed presence servers and server network topologies on the Internet, and propose an efficient and scalable server-to-server overlay architecture called Presence Cloud to improve the efficiency of mobile presence services for large-scale social network services.

we describe previous researches on presence services, and survey the presence service of existing systems. Well known commercial IM systems leverage some form of centralized clusters to provide presence services. Jennings III et al. presented taxonomy of different features and functions supported by the three most popular IM systems, AIM, Microsoft MSN and Yahoo! Messenger. The authors also provided an overview of the system architectures and observed that the systems use client-server-based architectures.

Skype, a popular voice over IP application, utilizes the Global Index (GI) technology to provide a presence service for users. GI is a multi-tiered network architecture where each node maintains full knowledge of all available users. Since Skype is not an open protocol, it is difficult to determine how GI technology is used exactly. Moreover, Xiao et al. analyzed the traffic of MSN and AIM system.

They found that the presence information is one of most messaging traffic in instant messaging systems. In, authors shown that the largest message traffic in existing presence services is buddy NOTIFY messages. Recently, there is an increase amount of interest in how to design a peer-to-peer SIP. P2PSIP has been proposed to remove the centralized server, reduce maintenance costs, and prevent failures in server-based SIP deployment.

To maintain presence information, P2PSIP clients are organized in a DHT system, rather than in a centralized server. However, the presence service architectures of Jabber and P2PSIP are distributed, the buddy-list search problem we defined later also could affect such distributed systems.

It is noted that few articles in discuss the scalability issues of the distributed presence server architecture. Saint Andre analyzes the traffic generated as a result of presence information between users of inter-domains that support the XMPP. Houri Show that the amount of presence traffic in SIMPLE can be extremely heavy, and they analyze the effect of a large presence system on the memory and CPU loading. Those works in study related problems and developing an initial set of guidelines for optimizing inter-domain presence traffic and present DHT-based presence server architecture.

Recently, presence services are also integrated into mobile services. For example, 3GPP has defined the integration of presence service into its specification in UMTS. It is based on SIP protocol, and uses SIMPLE to manage presence information. Recently, some mobile devices also support mobile presence services. For example, the Instant Messaging and Presence Services (IMPS) was developed by the Wireless Village consortium and was united into Open Mobile Alliance (OMA) IMPS in 2005.

In, Chen et al. proposed a weakly consistent scheme to reduce the number of updating messages in mobile presence services of IP Multimedia Subsystem (IMS). However, it also suffers scalability problem since it uses a central SIP server to perform presence update of mobile users. In, authors presented the server scalability and distributed management issues in IMS-based presence service.

2. DESIGN:
A) INPUT DESIGN:

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system.

The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

B) OUTPUT DESIGN:

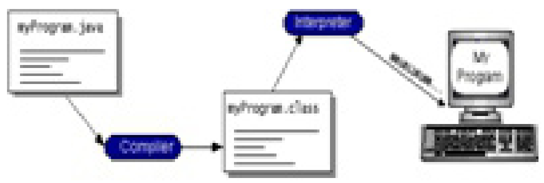
A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output.

It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements. Select methods for presenting information. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives. Convey information about past activities, current status or projections of the Future. Signal important events, opportunities, problems, or warnings. Trigger an action. Confirm an action.

3. Software Environment:

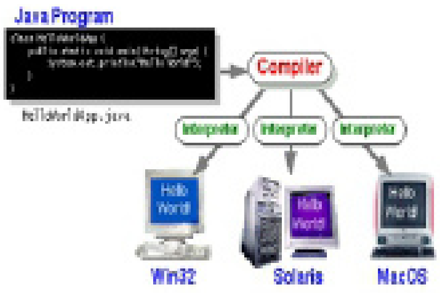
Java technology is both a programming language and a platform.With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works..



You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make "write once, run anywhere" possible.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

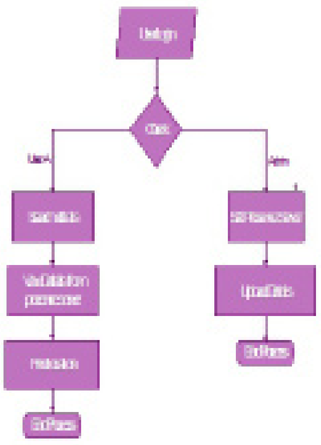


A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:
The Java Virtual Machine (Java VM) The Java Application Programming Interface (Java API)

Data Flow Diagram:

The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system.



4. MODULES:

1.Presence Cloud server overlay.

2.One-hop caching strategy.

3.Directed buddy search.

A) Presence Cloud server overlay :

The Presence Cloud server overlay construction algorithm organizes the PS nodes into a server-to-server overlay, which provides a good low-diameter overlay property. The low-diameter property ensures that a PS node only needs two hops to reach any other PS nodes.

B) One-hop caching strategy :

To improve the efficiency of the search operation, Presence Cloud requires a caching strategy to replicate presence information of users. In order to adapt to changes in the presence of users, the caching strategy should be asynchronous and not require expensive mechanisms for distributed agreement. In Presence Cloud, each PS node maintains a user list of presence information of the attached users, and it is responsible for caching the user list of each node in its PS list, in other words, PS nodes only replicate the user list at most one hop away from itself. The cache is updated when neighbors establish connections to it, and periodically updated with its neighbors. Therefore, when a PS node receives a query, it can respond not only with matches from its own user list, but also provide matches from its caches that are the user lists offered by all of its neighbors.

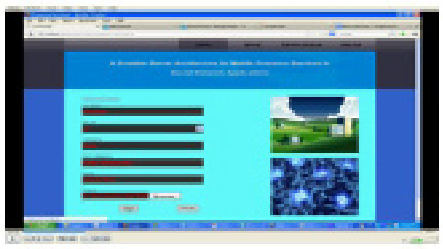
C) Directed buddy search :

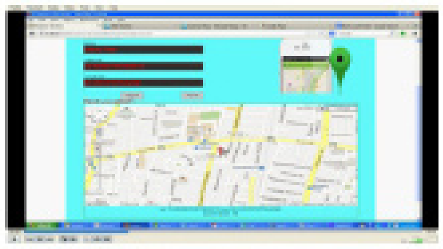
We contend that minimizing searching response time is important to mobile presence services. Thus, the buddy list searching algorithm of Presence Cloud coupled with the two-hop overlay and one-hop caching strategy ensures that Presence Cloud can typically provide swift responses for a large number of mobile users. First, by organizing PS nodes in a server-to-server overlay network, we can therefore use one-hop search exactly for queries and thus reduce the network traffic without significant impact on the search results.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Second, by capitalizing the one-hop caching that maintains the user lists of its neighbors, we improve response time by increasing the chances of finding buddies. Clearly, this mechanism both reduces the network traffic and response time. Based on the mechanism, the population of mobile users can be retrieved by a broadcasting operation in any PS node in the mobile presence service. Moreover, the broadcasting message can be piggybacked in a buddy search message for saving the cost.

DISCUSSIONS AND RESULT:





CONCLUSION

In this paper, we have presented Presence Cloud, a scalable server architecture that supports mobile presence services in large-scale social network services. We have shown that Presence Cloud achieves low search latency and enhances the performance of mobile presence services. In addition, we discussed the scalability problem in server architecture designs, and introduced the buddy-list search problem, which is a scalability problem in the distributed server architecture of mobile presence services.

Through a simple mathematical model, we show that the total number of buddy search messages increases substantially with the user arrival rate and the number of presence servers. The results of simulations demonstrate that Presence Cloud achieves major performance gains in terms of the search cost and search satisfaction. Overall, Presence Cloud is shown to be a scalable mobile presence service in large-scale social network services.

REFERENCES:

[1] R. B. Jennings, E. M. Nahum, D. P. Olshefski, D. Saha, Z.-Y. Shae, and C. Waters, ”A study of internet instant messaging and chat protocols,” IEEE Network, 2006.

[2] Gobalindex, http://www.skype.com/intl/en-us/support/user-guides/p2pexplained/.

[3] Z. Xiao, L. Guo, and J. Tracey, ”Understanding instant messaging traffic characteristics,” Proc. of IEEE ICDCS, 2007.

[4] C. Chi, R. Hao, D. Wang, and Z.-Z. Cao, ”Ims presence server: Traffic analysis and performance modelling,” Proc. of IEEE ICNP, 2008.

[5] Instant messaging and presence protocol ietf working group http://www.ietf.org/html.charters/impp-charter.html.

[6] Extensible messaging and presence protocol ietf working group http://www.ietf.org/html.charters/xmpp-charter.html.

[7] Sip for instant messaging and presence leveraging extensions ietf working group. http://www.ietf.org/html.charters/simplecharter.html.

[8] P. Saint-Andre., ”Extensible messaging and presence protocol (xmpp): Instant messaging and presence describes instant messaging (im), the most common application of xmpp,” RFC 3921, 2004.

[9] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle, ”Session initiation protocol (sip) extension for instant messaging,” RFC 3428, 2002.

[10] Jabber, http://www.jabber.org/.

[11] K. Singh and H. Schulzrinne, ”Peer-to-peer internet telephony using sip,” Proc. of ACM NOSSDVA, 2005.

[12] P. Saint-Andre, ”Interdomain presence scaling analysis for the extensible messaging and presence protocol (xmpp),” RFC Internet Draft, 2008.

[13] V. Ramasubramanian and E. G. Sirer, ”Beehive: o(1) lookup performance for power-law query distributions in peer-to-peer overlays,” Proc. of USENIX NSDI, 2004.

[14] A. Abdul-Rahman and S. Hailes., ”A distributed trust model,” Proc. of the workshop on New security paradigms, 1997.

[15] K.-T. Chen, C.-Y. Huang, P. Huang, and C.-L. Lei, ”Quantifying skype user satisfaction,” Proceedings of ACM SIGCOMM, 2006.

[16] P. Anick, ”Using terminological feedback for web search refinement: a log-based study,” Proceedings of ACM SIGIR conference on Research and development in informaion retrieval, pp. 88–95, 2003.