Instantaneous Messaging System: A Distributed Approach

Pablo S. García* **Omar E. Silva†** **Jorge R. Ardenghi‡**

Departamento de Ciencias de la Computación
Laboratorio de Investigación Sistemas Distribuidos
Universidad Nacional del Sur
Avenida Alem 1253 - (8000) Bahía Blanca - Argentina
Tel/Fax: (54)(291)4595135

Abstract

This work presents the development of an instantaneous messaging system, which is centered on the distributed handling of user groups. The method proposed for handling of groups has several distinctive characteristics: the non existence of a moderator for each group, allowing any user to abandon the group at any moment without putting in danger the group´s existence, and the absence of a server where the user must be authenticated every time that he wants to engage in conversation with somebody, which allows that a more direct communication be established with the person with which communication is desired.

Keywords:Distributed Systems, Groups of Communication, Instantaneous Messaging, Applications Intranet

1 Introduction

Communications, through computer networks, and specially Internet, are allowing different service alternatives. Users use e-mail, chatting, ICQ, videoconferences, IP telephony, and so on.

The existence of these alternatives delineates an evolution determined by the available technology, which has always been impelled by the prevailing necessity of communication. One of the characteristics that differs to the oldest of these systems as it is electronic mail. Of the others, it is the speed up in communications; although electronic mail allows the sending of messages, it is not possible to establish a communication in real time, while other systems offer this characteristic.

*garciapa@ns1.uns.edu.ar
†sindar@impsat1.com.ar
‡jra@cs.uns.edu.ar

Among the systems that allow a communication in real time it is possible to establish a difference between the systems of traditional chat and the rest; since the chat systems are designed to offer places where people meet, users choose this method if they want to communicate with others, but those systems don't offer the possibility of establishing a direct communication with other people.

On the other hand the systems of instantaneous messaging, telephony IP and videoconferences, allow direct communication between two or more users, being seen as to the operation of phone communications.

At first glance videoconference systems would lead to the others because they allow a more complete communication, in the sense that they allow to see and to hear the user with which one is communicating; however, these systems have disadvantages with regard to the instantaneous messaging, among which there is the fact that it is expensive and their quality is poor. As an advantage, the systems of instantaneous messaging are quicker in the communications and they require less resources to work.

The main interest of this work has been the instantaneous messaging systems because this type of system allows a real time communication between two or more people, allowing them to be linked to each other either from their work place or from their home. In this way instantaneous messaging plays a preponderant role as much in the business environment as in the personal environment.

Actually, there are two types of systems of instantaneous messaging, the publics and the private. Among the public are: Instant Messenger, AOL, ICQ, MSN Messenger, PowWow and Yahoo! Messenger. On the side of the private ones the ICQ stands out Groupware, Lotus Sametime and PowWow for Private Networks. While public instantaneous messaging is an option of low cost, this doesn't allow controlling of access to the service, an option that the private services offer.

Beyond the existent differences among these systems, they share the same characteristic: the existence of a server in which the list of the users of the system is stored. This characteristic gives it the possibility to inform a user, when a user of his list of contacts is on-line.

The proposed system of instantaneous messaging, thought to work on a LAN network where each machine has an IP address assigned (fixed or dynamic), as the previous ones, allows for real time communication between two or more people, but contrary to the others it doesn't use a server for the maintenance of lists of users or other data system. This form of operating, although it doesn't allow to determine if a given user belonging to the list of contacts is on-line or not, gives more robustness to the system since it doesn't depend on the correct operation of a server.

To be able to establish a communication in this system, the user should use the IP address or name of the host with which he wants to communicate.

The conversations that can be initiated within this system can occur between two people or between two or more people, establishing what is nominated *conversation group*.

Contrary other systems, a moderator doesn't exist for the conversation groups; every time a new participant wants to join to a conversation group, the responsibility of allowing him to enter or not, falls on the member of the group that received the

entrance order. This allows new members to enter or leave in a dynamic way, without any one of them should constantly remain in the group as moderator.

Before beginning with the description of the protocol on which the system is based, some aspects will be described.

The system described allows the communication between two or more users through conversation groups, which are managed in a distributed way; this implies the non-existence of a central server in which the users should register to be able to communicate among themselves.

Another distinctive aspect is the dynamic characteristic of the conversation groups, which exist while there are two or more users conversing in them and they cease to exist when it is a single user. This behavior contrasts with the traditional systems of chat, and it reflects the model in which is based, similar to a conversation or phone conference.

2 Protocol for Distributed Groups of Conversation

2.1 Components of the Protocol

To be able to describe the protocol it is necessary to specify a set of elements that are used in its operation.

The *conversation nodes* (nodes) are the ends among which the communications are established; these are carried out in the context of the denominated *conversation groups*.

A *conversation group* is a set of nodes that are communicating among each other, it does not have a name or global denomination; however it is represented in each node through a *group*, which has an explicit name that can be different in each node.

Each one of the members of a group is called a *participant*.

The participants communicate exchanging *messages* among each other.

Another important element of the protocol is the *coordinating node* (coordinator), this is a distinguished participant inside a conversation group, it centralizes and serializes the entrance of the other nodes in the conversation group.

2.2 Description of the Components of the Protocol

The main components of the protocol are the following:

Node: Besides being an end of a conversation, a node allows the creation, maintanence and the destruction of groups, as well as the establishment of communication with other nodes. Physically, each node has a direct association with a computer on the network, which has a unique IP id.

Conversation group: Regarding the cardinality of the conversation groups two types of groups exist. The *private groups* are those that only allow the establishment of a conversation between two nodes. The *public* groups admit two or more nodes.

Group: In the same way that the node maintains the groups, a group maintains the participants. A group is the place to which the messages arrive and from which the messages are sent by participants. Groups are discriminated by their members and through their type (public or private).

Participants: Each participant represents a different node inside a group and it is responsible for sending and receiving messages from and toward that node.

Messages: Two types of messages, the *control messages* and the *conversation messages* exist. The first type of message is used for the establishment and the termination of the communications, while the second one is the means through which the conversation is carried out.

Control messages: There are several control messages:

connection request: a message sent by a node that wants to establish a communication with another node of the network.

answer: a message sent by the node contacted to the node applicant of the communication. There are several types: **Yes**, **No**, **AnsweringMachine**, **Derivation**, **DisconnectedGroup**, **NoUpdatedGroup**.

connection release: a message sent by a participant when he wants to abandon a group.

automatic entering to the coordinator: a message sent by a node that wants to establish a communication with an existing conversation group of two or more nodes (public connection). As a consequence of the reception a message Derivation, a message of type answer is sent by a node to the coordinator. This message from the coordinator is always `Yes`.

automatic entering to peers: a message sent after receiving of the coordinator node the list of members of the conversation group to establish the connections with each one of them. The answer to this message can be `Yes`, `DisconnectedGroup` or `NoUpdatedGroup`.

list updating: a message sent by a coordinator of a conversation group to all the members of the conversation group after receiving a message of connection request or of automatic entrance to the coordinator, and before sending the list of participants to the new member.

Coordinator: It is the node which receive the request of others nodes for entering in the group that this node controls.

2.3 Operation of the Protocol

2.3.1 Description of the establishment of a conversation group (general case)

To create a conversation group between any two nodes, one of them should send the other one a message of connection request which indicates if the group is public or

private.

The node that receives this request can decide to reject it (message of answer type `No`) or to accept it; in this case the node should choose an existent group in which to overturn the conversation, or to create a new group for it. If it accepts the request the node sends an answer type `Yes`, and in the case of having chosen an existent group , the node also informs the members that are part of the group.

If the receiving node of the request chooses a group of which it is not the coordinator, the node sends an answer of type `Derivation` instead of an answer of type `Yes`.

If the received answer is of type `Derivation`, the node will try to communicate with the coordinator of the conversation group, choosing for it to the first member of the list with which a communication can be established, sending a message of automatic entering to the coordinator.

The coordinator responds automatically `Yes`, and he notifies to the participants of the group about the new participant's entrance, sending messages of list updating to each one of them.

Once the initial node receives the answer `Yes`, it will use the enclosed list of members to generate the group. In the case that it contains more than two members a message of automatic entering to peers will be sent to each one of them to be able to establish the communication with each one.

Once, the group in this node is created, its participants can begin to exchange conversation messages.

During their existence, the group will be able to grow or to diminish in its quantity of participants, disappearing to the point in which this quantity ends up being smaller than two participants.

Every time that a node wants to abandon a conversation group, it sends messages of connection release to each one of the nodes that are part of the conversation group.

For the sending of the conversation messages and connection release, a non reliable multicasting is used [Coulouris et al 2000], because one does not wait for an acknowledge for each message sent. This multicasting is not atomic.

3 Fundamental aspects of the protocol

The distributed operation of the conversation groups is sustained mainly o the coordinator's existence and the method of its election, which is designed to minimize the flow of messages in the network.

A more formal definition of group is: A group of n nodes is said to be a conversation group if: There only exists one connection among each couple of nodes. The number of connections among the n nodes is the combinatory number C_2^n .

The previous definition gives a static notion of conversation group, that is, a group in which there are no nodes trying to enter.

The node that wants to enter a conversation group should do so through an entrance operation to the group; this operation should guarantee that after its execution the group is in a consistent state.

A precondition for the execution of this operation is that the entering node is not already in communication, inside a group of the same type of the one that it wants to create (Private or Public), with some of the members of the group; that is, it is not allowed two nodes be in more than a group of conversation of the same type.

It is said that a conversation group reaches a consistent state after an entering node finishes establishing all the connections with all the previous members of the group. Understanding previous to mean all the nodes that were in the group, plus all those that tried to enter previously to the group and their entering was successful.

Every time that a node wants to enter a conversation group, it should do so through an entering operation to the group, which should leave the group in a consistent state; it is for this reason that it is possible to consider this operation a transaction, that is, this operation should take the group from one consistent state to another.

To be able to determine the previous members, at the moment of entering the conversation group, a coordinating node is necessary; which maintains an updated list inside the group (ordered by order of entering request) of all the nodes that are members or the nodes that want to enter the conversation group.

The coordinator allows the execution of the critical section of the entering transaction to the group in an atomic way (mutual exclusion).

3.1 Composition of the entering transaction to a conversation group

The entering transaction has three sections:

Entering section: from the moment the message of connection request is composed until before the message arrives to the coordinator (it is possible that the message has become a message of automatic entering, previous derivation).

Critical section: from the moment the coordinator receives the request message or automatic entering, until the coordinator sends the answer message `Yes` (previously, it is sent to all the nodes members of the group the message of list updating).

Exit section: from the moment the node receives the answer `Yes` from the coordinator until it finishes establishing the connections with the members of the list sent by the coordinator (*piggybacked* with the answer `Yes`).

Considering that the execution of the critical section is carried out atomically, any concurrent planning of the sections (and still separating the entrance sections and exit) will be serializable, that is, equivalent to a serial planning.

Once the answer `Yes` on the part of the coordinator is received, the node should establish the connections with each one of the members of the enclosed list with the answer. To carry this out, the node sends messages of automatic entering to peers to each member; in the case of not being able to establish communication with some of them, it simply eliminates them from the list (if a node is disconnected for some reason

from the group, each one of its members will be notified of eliminating it from the list of members of the group).

After sending the messages of automatic entering to peers, the node waits for a time determined by the corresponding answer of each one of the members of the group; these answers can be: `Yes`, `DisconnectedGroup`, `NoUpdatedGroup`.

After having finished the waiting, the answers of the other nodes of the group are analyzed, if some of the nodes have answered `NoUpdatedGroup` the entrance operation to the group will not be successful, and therefore messages of connection release will be sent to each one of the contacted nodes so they can update their lists and assure the atomicity of the transaction.

If all the answers were `Yes`, `DisconnectedGroup`, or a combination of both; the entrance operation to the group finishes successfully.

3.2 Method of the coordinator's election

Before describing the method, it is important to clarify that it is supposed that, when for some problem in the network a node is disconnected, it is detected by all the other nodes of the conversation group, generating in each one of them the elimination, in the list of members, of the node that was disconnected.

The method of the new coordinator's election requires the existence, in each group of each node, of the list of members of the conversation group; which should have the members of the group in the same order, in each one of the nodes.

Once these conditions are satisfied the method consists of choosing as new coordinator the first member of the list; if the current coordinator (first member of the list) abandons the group, after eliminating it from the list, the member that continues in order, in the list will be new coordinator , and will be connected to the conversation group.

It is necessary to highlight that contrary to the traditional election methods (*Bully Algorithm*, *Ring Algorithm* [Coulouris et al 2000, Garcia Molina 1992, Singh et al 1994, Fredrickson et al 1987], this algorithm doesn't require that the system enters in a state in which the only operation that it is carried out is that of election and where it should remain until the algorithm concludes; since the election is carried out in an independent way in each node without the necessity of passage of messages among them. This characteristic makes it appropriate for supporting the dynamic of the conversation groups.

Comparing, [Coulouris et al 2000], describes the implementation of a service for groups of processes, which uses several concepts and similar mechanisms. It can also be [8] a simplified model of communication among peers applied to a messaging system.

It becomes difficult to establish a precise comparison among the protocol described in this work and the protocols of the commercial systems, because these are not published. However, in the RFC 2778 (TO Model for Presence and Instant Messaging. M. DAY, J. ROSENBERG,H. SUGANO. February 2000) it is possible to find the description of a model which is comparable with this work.

4 Conclusion and Future Works

In this work a description of a system of instantaneous messaging is presented; this system allows the establishment of a direct communication between two or more users (using the IP address or name of the machine where the user to be connected is), carrying out, also, a distributed handling of the conversation groups avoiding in this way the use of a server for the maintenance of all the users of the system (each user should be authenticated at the moment of establishing a communication).

This system uses the communication in groups of peers through a non reliable multicasting for the interaction among the members.

Currently an implementation of the system, written in JAVA, is in test stage . This language was used due to its portability.

To establish the communications, the sockets provided by Java are used. These sockets work on TCP/IP which allows the correct operation of the protocol; since TCP is a connection oriented protocol, it guarantees that the delivery of the messages is made in the same order in which they were sent, being this a fundamental requirement for the operation of the method of the new coordinator's election.

To eliminate certain limitations of the system, extensions that should be made in future works are proposed:

Implementation of a list of contacts with information regarding the user's name and the IP addresses where that user can be found.

Development of a server for the maintenance of a list of users that voluntarily decide make their name and IP address available so that other users can communicate with them. This server should also be used by the user, if he wants, to inform by this way of his on-line presence. It is necessary to highlight that the existence of this server is not a necessary condition for the operation of the system.

Development of a proxy server that supports the protocol of the system and allows the users behind the proxy to communicate with external users to the proxy and vice versa.

Implementation of message encryption.

References

[Birman et al 1993] K. P. Birman, T. Joseph, *The Process Group Approach to Reliable Distributed Computing,* Communications of the ACM, vol.36, pp.36-53, Dec. 1993

[Comer 1995] D. Comer, *Internetworking With TCP/IP Volume 1: Principles Protocols, and Architecture,* 3rd edition, Prentice Hal, 1995.

[Comer 1997] D. Comer, *Internetworking With TCP/IP Volume III: Client-Server Programming and Applications, BSD Socket Version,* 2nd edition, Prentice Hall, 1997

[Coulouris et al 2000] G. Coulouris, J. Dollimore, T. Kindberg, *Distributed Systems: Concepts and Design,* 3rd edition, Addison-Wesley, 2000

[Fredrickson et al 1987] N. Fredrickson, N. Lynch, *Electing a Leader in a Synchronous Ring* Journal of the ACM, vol.34, pp.98-115, Jan. 1987

[Garcia Molina 1992] H. Garcia Molina, *Elections in a Distributed Computing System,* IEEE Trans. on Computers, vol.31, pp.48-59, Jan. 1992

[Lockhart 1994] H. W. Lockhart, *OSF DCE Guide to Developing Distributed Applications,* Mc. Graw Hill, 1994.

[Silberschatz et al 1998] A. Silberschatz, P. Galvin, *Operating System Concepts.* 5th edition, Addison-Wesley.1998

[Silberschatz et al 1999] A. Silberschatz, H. Korth, S. Sudarshan, *Database Systems Concepts.* 3rd edition, McGraw-Hill, 1999.

[Singh et al 1994] S. Singh, J. Kurose, *Electing Good Leaders,* Journal of Parallel and Distributed Computing, vol.21, pp.184-201, May 1994

[Tanenbaum 1996] A. Tanenbaum, *Computer Networks,* 3rd edition, Prentice Hall, 1996