Demo Abstract: sChat: A Group Communication Service Over Wireless Sensor Networks

Fei Sun, Chien-Liang Fok, Gruia-Catalin Roman
Dept. of Computer Science and Engineering
Washington University in Saint Louis
Saint Louis, MO, 63105, USA
{fs2, liang, roman}@cse.wustl.edu

Categories and Subject Descriptors: C.2.2 [**Network Protocols**]: Applications; C.3 [**Special-Purpose and Application-based Systems**]: Real-time and embedded systems

General Terms: Algorithms, Design, Experimentation

Keywords: Wireless Sensor Network, Mobile User, Group Communication, Multicast

1. INTRODUCTION

Wireless sensor networks (WSNs) consist of a collection of low power devices called *motes*, and have been deployed in many applications [6]. As the technology matures and more applications become feasible, WSNs may one day be ubiquitous. This will form a resilient sensing and communication infrastructure because WSNs are typically ad hoc, wireless, and self powered. Thus, in a disaster scenario where traditional communication infrastructures like cell phone towers are unavailable, WSNs will likely continue to work. This can serve as an alternative medium through which groups of people like emergency crews and family members can communicate. Facilitating this, however, requires developing a new WSN group communication service.

In our demo, we present sChat, a group communication service for WSNs. sChat allows groups of mobile entities (e.g., rescuers) to communicate over a WSN. In sChat, mobile users can access the WSN using portable wireless devices. Once these mobile users connect to the network, they can discover existing groups and then either join one or create a new one. Members in each group are able to send and receive messages to and from all the other group members, even if they are physically far from each other. Members can also leave groups at any time. The groups are formed on demand, and are disbanded when all the members are left.

Since people move, sChat must address challenges introduced by mobility like ensuring all group members stay in touch despite unpredictable member movements or random disconnections. sChat uses a leader in each group to keep track of member locations and distribute group broadcast messages. When the mobile user moves, the leader is notified with the new location, ensuring that the member continues to recieve group broadcast messages.

We assume that WSNs are initially deployed for some other purpose, meaning a group communication service is not pre-built into the network and has to be added. In order to provide our sChat service, we need a software update mechanism that allows us to introduce new code into the network. Numerous technologies have been developed for this purpose [5, 1, 2]. For this demo, sChat is developed on one of these technologies, Agilla [2], a mobile agent middleware for WSNs.

IPSN'07, April 25-27, 2007, Cambridge, Massachusetts, USA.
ACM 978-1-59593-638-7/07/0004.

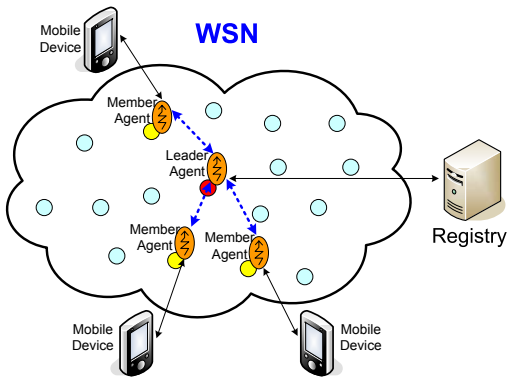

Figure 1: The sChat system architecture. Users in the same group carry mobile devices that interact with member agents which communicate through a leader agent. The leader agent is registered with the registry and distributes group messages.

2. IMPLEMENTATION

In sChat, each group is identified by a unique name and is formed when the first member requests to join it. Each group has a leader, which is hidden from the mobile users. This distinguishes sChat from other group communication protocols [3, 7]. The leader remembers each member's location, and is responsible for distributing group messages among the members. In addition, since members are mobile, the leader must continuously update the member location records to guarantee message delivery. The leader should also adjust its location to minimize group communication overhead. When the last member leaves the group, the group is disbanded. A registry, e.g. a base station, keeps track of all the groups, their names, and their leaders' locations. Figure 1 provides an overview of sChat's system architecture. It consists of users carrying mobile devices that interact with member agents within the WSN. These

member agents serve as proxies for the users, and communicate with each other through a leader agent. The registry is updated every time the leader changes location.

sChat provides the following three operations: join group, send message, and leave group. Each of these are now described in more detail.

Join Group: When a member joins a group, the service first queries the registry with the group name. If the group name is registered at the registry, meaning the group already exists, the registry will respond with the corresponding leader's location. If a group does not exist when a member joins, the registry will create a new group by recording the group name, injecting a new leader for the group (initially at the first member's location), and respond with the new leader's location. After the member gets the leader location, it will send a *join group* message to the leader containing its current location. Upon receiving this join message, the leader will add the member and its location to its member list, concluding the join process.

Send Message: When a member wants to send a message to the group, it first sends the message to the leader, and the leader broadcasts the message to the members. All messages exchanged within the group must go through the leader.

Leave Group: When a member wants to leave a group, it will send a *leave group* message to the leader. The leader will then remove the member from its member list. Once the last member leaves the group, the leader sends a *disband group* message to the registry, which deletes the group's record.

User mobility can potentially cause a member to disconnect from the group. sChat handles this by updating a member's location with the leader after the member moves. When a member arrives at its new destination, it sends the leader an *update location* message, and the leader updates its records, ensuring that the member remains part of the group. If many members move, the leader may be located far away from the group members, resulting in high communication overhead [4]. In our design, the leader is ideally at the centroid of all the group members. If the members move such that the distance between their centroid and the leader location is above a certain threshold, the leader will move to the centroid. When the leader moves, it will update its location with all the group members and the registry.

sChat is implemented on Agilla, a mobile agent middleware for WSNs. Mobile agents are special software processes that are capable of migrating across motes. Mobile agents are used to represent group members and leaders within the WSN. Messages are exhanged through Agilla's tuple spaces, and the routing is done through Agilla's built-in geographic routing.

3. DEMONSTRATION

In our demonstration, we will show how sChat enables mobile users to chat in groups via a WSN. We will setup a WSN consisting of 40 Tmote Sky motes in a 8x5 grid layout. All motes will run TinyOS and Agilla. We use a laptop as the registry to keep track of the groups in the network. We provide portable wireless devices like Sumsung's Origami Ultra-Mobile PC with telosb motes attached to them so that they can interface with the WSN. Visitors can use these portable devices to group chat over a WSN. Currently, sChat only supports text messaging due to bandwidth constraints. In the future, our service may be easily extended to support audio and video communication.

When a user starts the sChat application, a GUI is created and a member agent is injected onto the mote that is closest to the user in the WSN. This agent serves as a proxy for the user within this WSN. Before users can start chatting, the GUI will prompt the user asking which group they want to join. Once the users join a group, they can start chatting via the WSN.

While chatting, the user can move around the network and still be able to send and receive messages. In a real scenario, when a user moves, he or she comes within range of different motes. Due to space constraints, in this demo, users model movement by explicitly telling the GUI where they are. When a user wants to leave a communication group, he or she needs to notify the service by clicking a *leave group* button on the GUI. This will cause the member agent to deregister with the group leader and then die, freeing up its resources for other users.

The registry will run a graphical display in real time, showing a map of the WSN and all existing communication groups. Through this, visitors can visualize the locations of the mobile users and the group leaders in the WSN, and see the groups being formed and disbanded.

Acknowledgment

This work is funded by the NSF under NOSS grant CNS-0520220.

4. REFERENCES

[1] A. Boulis, C.-C. Han, and M. Srivastava. Design and implementation of a framework for efficient and programmable sensor networks. In *Proc. of MobiSys'03*, pages 187–200. USENIX, May 2003.

[2] C.-L. Fok, G.-C. Roman, and C. Lu. Rapid development and flexible deployment of adaptive wireless sensor network applications. In *Proc. of ICDCS'05*, pages 653–662. IEEE, June 2005.

[3] J.-H. Huang, J. Buckingham, and R. Han. A level key infrastructure for secure and efficient group communication in wireless sensor network. In *Proc. of SECURECOMM'05*, pages 249–260, Washington, DC, USA, 2005. IEEE Computer Society.

[4] H. S. Kim, T. F. Abdelzaher, and W. H. Kwon. Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks. In *Proc. of SenSys'03*, pages 193–204, New York, NY, USA, 2003. ACM Press.

[5] P. Levis and D. Culler. Maté: a tiny virtual machine for sensor networks. In *Proc. of ASPLOS-X'02*, pages 85–95, New York, NY, USA, 2002. ACM Press.

[6] K. Romer and F. Mattern. The design space of wireless sensor networks. *IEEE Wireless Communications*, 11(6):54–61, 2004.

[7] B. rong Chen, K.-K. Muniswamy-Reddy, and M. Welsh. Ad-hoc multicast routing on resource-limited sensor nodes. In *Proc. of REALMAN'06*, pages 87–94, New York, NY, USA, 2006. ACM Press.