Demo Abstract: A Hierarchical Location Directory Service Across Sensor and IP Networks

Sangeeta Bhattacharya, Chien-Liang Fok, Chenyang Lu, Gruia-Catalin Roman
Department of Computer Science and Engineering
Washington University in St. Louis
{sangbhat,liang,lu,roman}@cse.wustl.edu

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.3 [**Special-Purpose and Application-Based Systems**]: Real-time and embedded systems

General Terms

Management, Design, Experimentation, Performance

Keywords

Sensor Networks, Location Directory, Mobile Agent

1 Introduction

Harry Potter used his "Marauders Map" to keep track of and locate wizards and magical creatures in the Hogwarts School of Witchcraft and Wizardry. Is it possible that we muggles (humans) can also have a marauders map that would help us keep track of mobile entities? The answer is "Yes indeed." Sensor networks have long been used to track mobile entities and much work already exists on tracking using sensor networks. However, existing systems only track mobile entities within a single sensor network.

Consider the example of co-ordinating doctors over multiple make-shift clinics, set up after a natural calamity. Such make-shift clinics are often short of doctors and so there may be a few doctors working round the clock and shuttling between the various clinics, depending on the need of the clinics. In such a scenario, there is often a need to keep track of the location of the doctors, as they move between clinics, so that it is possible to find the nearest available doctor in case of a medical emergency or to locate a particular doctor or to just know the number of doctors at a particular clinic. In order to support this and other similar large-scale location-based applications, it is essential to provide a location directory service that can efficiently keep track of mobile entities across *multiple sensor networks*. Moreover, the directory service should be able to answer a broad range of *spatial queries* that may span multiple sensor networks and may be issued from *any point* in a sensor network.

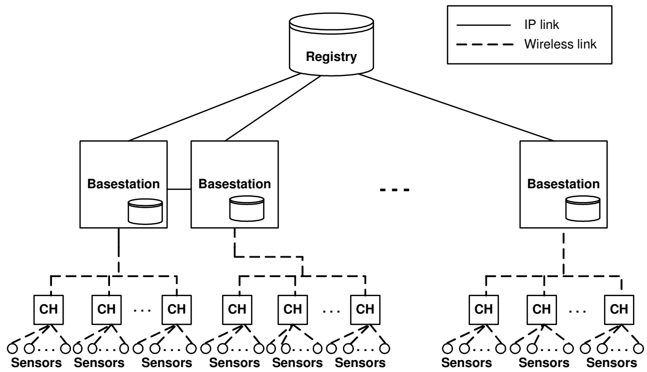

Figure 1. **MADS Architecture.**

We have thus developed MADS, a location directory service that meets the above requirements and hence enables the realization of wide-area, location-based applications such as city-wide public surveillance systems, and regional rescue operations after disasters. MADS is a directory service specifically for mobile agents, which represent mobile entities in the physical world (e.g. cars, people, etc). Its key features are as follows:

- It maintains location information of mobile agents as they move within and between sensor networks.
- It supports a broad range of spatial queries concerning the mobile agents, which may be limited to a single sensor network or may span multiple sensor networks.
- It supports queries of multiple granularities, allowing flexible tradeoff between location accuracy and communication cost.
- It has very low communication cost.
- It is highly scalable with respect to network size and the number of mobile agents that it can keep track of.

MADS is designed for a system of static sensor networks, each of which consists of static location aware sensor nodes and a stationary base station. The base stations of the different sensor networks are assumed to form an IP network, thus connecting the sensor networks.

MADS has a hierarchical architecture, based on this view

SenSys'06, November 1–3, 2006, Boulder, Colorado, USA.

of tiered sensor networks. The hierarchical architecture also enables MADS to achieve scalability, low communication cost and multiple query granularities. The four-tiered hierarchical architecture of MADS is shown in Figure 1. As shown in the figure, sensor nodes form the lowest level (level 3) of the hierarchy. The sensor nodes are organized into clusters, with each cluster having a clusterhead. The clusterheads form level 2 of the hierarchy and report to a base station. The network of base stations belonging to different sensor networks form level 1 of the hierarchy. The base stations register their IP addresses, network names and the agents in their network in a registry, which forms level 0 of the hierarchy. Note that the entire system consists of heterogeneous nodes with those at higher tiers having more resources than those at lower tiers.

MADS employs a distributed directory structure and stores data at different resolutions, at different levels of the hierarchy. The clusterheads maintain the exact location of the mobile agents that are in their cluster, while the base stations maintain a list of mobile agents that are in their sensor network along with the cluster that the agents are in. Finally, the registry maintains a list of mobile agents in the system of sensor networks along with the networks that the agents are in. This distributed directory structure minimizes communication cost and allows MADS to support different query granularities.

2 Implementation

We have implemented MADS as a middleware service and have integrated it with Agilla [1], a mobile agent middleware for the TinyOS platform. Agilla provides primitives for the creation of mobile agents that display application-specific behavior. Once injected into the network, mobile agents can move and clone themselves across sensor nodes.

Currently, MADS supports four types of spatial queries and allows the application to specify the scope (S) and the desired granularity (G) of each of these queries. The scope (S) of a query can be either *local* or *global*. Local queries are limited to the local sensor network while global queries are based on information from across multiple sensor networks. Global queries return the same information as local queries but also return the network ID. The granularity (G) of a query can be either *fine*, *coarse* or *network*. The result of a fine query is based on the exact locations of the mobile agents, while the query result of a coarse query is based on the knowledge of the clusters that the agents are in. The result of a network query, on the other hand, is based only on the knowledge of the networks that the agents are in. Thus, the query result varies according to the scope and granularity of the query. In addition to the parameters *S* and *G*, some queries also take in the parameter *C* that specifies the "class" of a mobile agent. This allows the application to search for agents within one class or in all classes. For example, if mobile agents are used to represent doctors and the mobile agent class denotes the doctor's specialization, then MADS can handle queries limited only to "surgeons" or to all doctors. The four spatial queries supported by MADS are summarized as follows:

1. **GetLocation(*id*, S, G)**: Get location of an agent with ID *id*.
2. **GetAllAgents(C, S, G)**: Get the location of all agents of class *C*.
3. **GetNumAgents(C, S, G)**: Get the number of agents of a particular class *C*.
4. **GetNearestAgent(C, L, S, G)**: Get the location of an agent of class *C* that is nearest to the location *L*.

3 Demonstration

We demonstrate MADS' ability to keep track of mobile agents as they move across sensor networks as well as its ability to support a wide range of spatial queries of multiple granularities. In order to demonstrate this, we emulate the earlier example of the make-shift clinics. We assume that there are two clinics, each covered by a sensor network, consisting of 12 nodes arranged in the form of a 4×3 grid. Two laptops, connected by an ethernet cable, are used as base stations for the two networks. Mobile agents are used to represent doctors who move about randomly within a clinic and across clinics. Once injected into a sensor network, the mobile agents move about randomly in the network. The agents randomly cross over to the other sensor network and then move about randomly in the other sensor network. We use Agimone [2] to enable a mobile agent to move from one sensor network to another, via the IP link between the two networks. The presence of a mobile agent in the sensor network is indicated by turning on the red/green/yellow LED of the sensor node hosting the mobile agent. Thus, the actual movement of the mobile agents in the network can be observed by looking at the LEDs of the sensor nodes. We also display the location and class of the mobile agents, as maintained by MADS, via a GUI on the laptops.

This setup is used to realize the following scenarios: (1) Display the location of all doctors, as they move around in the clinics, via the GUI on the laptops. This scenario makes use of the **GetAllAgents** query to track the mobile agents representing the doctors, as they traverse a sensor network and move between sensor networks. Here, we use only a *coarse* query, to show the approximate location of the doctors. This experiment is aimed at demonstrating MADS' ability to keep track of mobile agents across multiple sensor networks. (2) Find the doctor closest to the doctor issuing the query. This scenario uses the **GetNearestAgent** query, with *fine* query granularity. (3) Find the number of surgeons (represented via agent class) in each clinic. This uses the **GetNumAgents** query at the *global* level, restricted to agents of the "surgeon class".

Acknowledgement This work is funded by the NSF under the ITR grant CCR-0325529 and the NOSS grant CNS-0520220.

4 References

[1] C.-L. Fok, G.-C. Roman, and C. Lu. Rapid development and flexible deployment of adaptive wireless sensor network applications. In *ICDCS'05*.

[2] G. Hackmann, C.-L. Fok, G.-C. Roman, and C. Lu. Agimone: Middleware support for seamless integration of sensor and ip networks. In *DCOSS'06*.