

# Middleware Design for Integration of Sensor Network and Mobile Devices\*

Vladimir Dyo

Department of Computer Science, University College London  
Gower Street, London, WC1E 6BT, UK

v.dyo@cs.ucl.ac.uk

## ABSTRACT

Integration of sensor networks with mobile devices can provide additional flexibility and functionality for a variety of applications and can have a significant practical potential. Designing applications for such integrated networks is a difficult task. The paper discusses the key research issues associated with such integration and our approach to solve these problems by designing a middleware architecture for integration of sensor networks with mobile devices.

## Categories and Subject Descriptors

C.2.4 [Computer-Communication Networks]: Distributed Systems—*Distributed Applications*; H.3.4 [Information Storage and Retrieval]: Systems and Software—*Distributed Systems*

## General Terms

Algorithms, Performance, Design

## Keywords

Wireless Sensor Networks, middleware

## 1. BACKGROUND AND MOTIVATION

Sensor networks consist of tiny low-powered computing devices with extremely restricted computational, communication and battery capabilities. Each device may be equipped with a physical sensor for reading temperature, sound, pressure or other physical phenomena and can operate both as a

sensor and a wireless router. One of the major tasks of sensor networks is the distributed collection and processing of sensor readings over extended periods of time [1]. Scalability, self-configuration, ease of deployment and low cost have made sensor networks very attractive solution for a wide range of environmental monitoring, distributed surveillance, healthcare and control applications.

Let us consider a scenario where sensors are used for a large scale environmental monitoring. Thousands of sensors are installed on trees, houses, along the roads and form a large scale multi-hop sensor network. The information coming from sensors is used for climate and air pollution monitoring. Each sensor is measuring temperature, humidity, air pollution or some site-specific phenomena of interest. The data generated by sensors has to be somehow *collected* and analyzed either in a centralized or distributed way. The traditional approach where data is collected at one or several centralized points is neither energy-efficient nor scalable [9]. One of the possible approaches to collect data from such a large scale network is through the use of mobile collectors (installed on robots, cars or people). This approach has a number of advantages over the centralized approach:

- Energy efficiency. Energy is the most critical factor when designing sensor networks. Measurements suggest that sending 1bit is equivalent to performing approximately 1000 CPU instructions. Therefore, any solution devised for sensor networks has to minimize the amount of communication overhead. In a mobile user scenario sensors will be storing data locally and provide it at request to mobile collectors. The data can be sent directly to a collector in 1-hop communication which reduces the need for multi-hop communication. Alternatively a mobile collector can move closer to the place where data is located which will reduce the number of message retransmissions.
- Extended coverage of sensor network and allows to access remote and disconnected parts through Wi-Fi or cellular interface. Some parts of the network can be just partially connected and central data collection could be impossible. The network can also become segregated in the course of operation as some sensors run out of power and die. In this situation, mobile collectors will be particularly helpful as the mobile users can physically move towards the location of data and collect it.

\*"Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. 2nd International Doctoral Symposium on Middleware '05, November 28- December 2, 2005 Grenoble, France Copyright 2005 ACM 1-59593-267-4/05/11... \$5.00"

- Cost. Using this approach there is no need for centralized infrastructure. Setting up multiple Data Collection points in different parts of sensor network could help but it does not solve the problem in principle. Also, it can be quite expensive if each data collection point has to be connected to a backbone (either through wired or wireless interface).
- Mobility of users. Users can query the network and collect data from any location in sensor network.

It is easy to see that our scenario applies to a wider range of applications. We are also investigating the application of sensors for a wildlife conservation system where a static sensor network is used to collect movement patterns of animals tagged with RFID sensors. This application could significantly help zoologists to study and protect endangered species.

The central point in the scenario are the mobile data collectors which perform dual role: they collect data and transfer messages between individual sensors. In many situations, similar to those mentioned in the scenario, collecting data at a certain fixed location is neither possible nor practical. Having mobile collectors can be the only way to solve the problem.

Developing applications for such scenarios is a difficult task. The primary reasons for this are device heterogeneity and the difficulty of low-level programming of embedded devices. In our scenario, applications have to be developed for various sensor platforms and be integrated with other mobile devices such as laptops, PDAs, mobile phones and sensors. As a result developers spend their time dealing with hardware and software peculiarities of a particular platform rather than concentrating on the application logic. Programming in NesC language, for instance, can be difficult and prone to errors. Secondly, introducing mobility to data collectors raises a number of new algorithmic problems. There is little research on data collection protocols for mobile sinks. Even though the area of hybrid sensor networks has been actively investigated by robotics community there is not much work in the network and middleware community. We have managed to find only few works on routing and data collection for mobile sinks [3] [5] but they focus on specific issues only such as multicast or calculating throughput.

After conducting an extensive literature review we have identified three unsolved major problems coming from the scenario where mobile users interact with a sensor network:

- It will require efficient data indexing algorithms for sensor network. A user does not need to collect all data from sensor network. Sometimes it might not even be possible, because of the amount of data the network can generate. It can be more energy-efficient to store the measurements locally and provide them to a user at request. This would require energy-efficient data indexing algorithms, so that mobile users could always query the network for interesting data. The indexing algorithm has to be adaptive to data event and incoming query rates. For example, if a local area

does not receive any queries, the area should not be indexed. Or alternatively, if the data is too dynamic and constantly changing, in this case it might be more efficient to index data on the fly, in response to an incoming query.

- It will require faster and better data aggregation and data collection algorithms as mobile users cannot wait for data to aggregate and arrive. Moreover, the amount of data can be more than a mobile user can collect while moving through the sensor network. Therefore, data collection techniques should take into account user mobility, speed and the channel bandwidth between user and sensor network. Sensors have to be able to send data to a mobile device at any moment of time. Because the location of a mobile device is not known, this will require either a) broadcasting entire network to find sink's location or b) constantly track the location of a sink. Both methods require communication overhead and the efficiency might be context dependant. It is not clear at this moment which method is more energy-efficient. The techniques for tracking devices will depend on user mobility, number of users and other factors.

Mobility of users (collectors) adds additional flexibility and complexity to the system. The data can now be collected at any place in the network. However, this mobility comes at a cost because the sensor network either has to be constantly reconfigured to track a mobile user thus creating additional communication overhead. Or a mobile user will always have to query the static location which also adds overhead. Our first research question is: what are the performance penalties and advantages for introducing user mobility for data collection in static sensor networks?

We believe that there is a significant gap in research describing interaction of mobile devices with static sensor networks. Our hypothesis is therefore twofold: a) a specialized data-collecting protocol for mobile users will be more energy-efficient than existing data-collection protocols. b) it is possible to design a middleware which allows interaction between mobile users and sensors.

## 2. EXPECTED CONTRIBUTION

The major contribution of this project is to design middleware for data retrieval applications with mobile data collectors. The middleware should provide application developers with higher level programming abstractions, which would allow developers to concentrate on the logic of the application rather than on implementation of low-level functionalities. This should facilitate software development and reduce development costs.

### 2.1 Middleware Architecture

Traditionally, the goal of the middleware has been to provide a set of programming and communication abstractions to facilitate the software development for heterogeneous systems [7]. In mobile computing middleware is also responsible for context awareness and adaptation [4]. The abstraction and generalization usually come at the expense of efficiency. This creates two major problems for sensor networks.

Sensors are extremely limited in their resources especially battery power. In sensor networks where the efficiency is of utmost importance the applications are usually "cross-layer" integrated and implement all the functionalities across the networking stack. Such functions as network access, adaptation, security and transparency can be implemented and tightly interweaved within one application making abstraction undesirable. For example, Directed Diffusion relies on its own mechanism to deliver data. Data-centric storage relies on geographic routing algorithms such as GPSR [11] to deliver data to a specific destination. Secondly, sensors are performing a highly specialized task, e.g. environmental monitoring, location tracking, healthcare applications and many others; therefore providing programming abstractions for all types of applications might prove to be difficult and not efficient. Programming abstractions come at a cost of performance, abstractions might be too general and have functionality not needed by the application. Our research question is a trade-off between abstraction and performance in sensor networks where performance is of utmost importance.

The primary part of the contribution would be designing architecture and the abstractions to express the functional and non-functional requirements of mobile sensor applications in a transparent and application independent way. Based on the requirements we will develop middleware primitives for data collection applications.

Designing middleware is a huge task, and taking into account limited time and resources We will focus on the two aspects of this middleware: middleware architecture and on the data collection algorithm. The primitives have to be mapped to protocols and algorithms used for communication and data retrieval. More discussion about the protocols is in the following subsection.

## 2.2 Data Retrieval Protocols for Mobile collectors

This section describes our recent work on how sensor network can serve continuous queries from mobile users in the most energy-efficient way. Mobile users travelling through the network will be interested in spatial queries asking for locations of specific data. The first question a mobile data collector is going to ask is whether sensor network has any interesting data available, and if yes, where exactly it is located. A typical query for this scenario could be "What are the locations of all critical events within a range  $R$  from a point with coordinates  $(X, Y)$ ?" This information could be used by a mobile user not only to collect data but also to navigate through geographically vast areas. Flooding a query each time across entire sensor network is not always desirable, especially in large scale networks as it creates significant communication overhead and drains too much energy.

Answering these questions would require a distributed index of entire network. Given the variability of the data and query rates, an index would need to be dynamically reconfigured. We consider a special case where a sensor network is receiving continuous queries from mobile collectors. The problem of energy-efficient data retrieval and distributed indexing in sensor networks have been addressed in

more detail in previous work [3] [10]. In general, there are two techniques that help creating and maintaining an index infrastructure: what we call a proactive approach and a reactive one. In proactive mode, sensors periodically report all events to a cluster head. A cluster head maintains up-to-date information about all events in the area and provides a quick response to lookup queries. This approach has some drawbacks: excessive communication overhead in the case of dynamic data sources; also, an index has to be maintained even when there are no lookup queries. In reactive mode data is pulled from sensors only in response to an incoming query. However, because the query has to be flooded over the network the cost of each query can be prohibitively high, so this technique is recommended only when the data is highly dynamic and the proactive approaches would not perform well. In [6] we have shown that using a combination of proactive and reactive modes for each sensor can minimize the communication overhead.

Once the mobile user identified the locations of all interesting data in the network it has to somehow collect all the data in the most efficient way. We are currently working on designing more efficient data collection protocol for mobile collectors. This scenario will require faster and better data aggregation and data collection algorithms as mobile users cannot wait for data to aggregate and arrive. Moreover, the amount of data can be more than a mobile user can collect while moving through the sensor network. Therefore, data collection techniques should take into account user mobility, speed and the channel bandwidth between user and sensor network. Sensors have to be able to send data to a mobile sink at any moment of time. Because the location of a mobile sink is not known, this will require either a) broadcasting entire network to find sink's location or b) constantly track the location of a sink. Both methods require communication overhead and the efficiency might be dependant on such factors as mobility and number of users and network density. More research is needed which approach to take depending on these and other factors.

Surprisingly, there is little research dedicated to data collection using mobile sinks. We have managed to find several works, and some relevant works from the areas of ad hoc routing and location tracking. The scenario where several mobile users need to collect data from fixed locations in sensor network has been investigated by Bhattacharya et al [3]. The authors proposed a data collection algorithm which creates a minimum Steiner tree connecting all mobile sinks and data locations. The tree also creates local caches so that data from a source to several sink is served from that cache. Their algorithm minimizes the total amount of communication overhead to feed data from one source to several mobile observers. The authors do not investigate such issues as data aggregation and compression considering them highly application specific tasks. In [5] Chakrabarty et al study the problem of maximum data throughput from multiple sensor networks to a mobile sink. They model data streams from individual sensors as Poisson events and use queuing theory to derive throughput. Their data collection techniques take advantage of predictable user mobility of the observer to reduce communication range and therefore power consumption for sensor network. A research group led by professor Hubaux has initiated a project of routing

towards mobile sinks in sensor networks. The status of the project is not known and the group has not published any results yet. Lamarca et al [12] mention that using mobile sinks can save energy and make sensor network more practical. However, they do not investigate how it could be done. Therefore more research is needed to investigate what benefits a mobile sink can give to a sensor network.

The area of hybrid sensor networks has been actively investigated by robotics community. In [2] mobile robots deploy static sensors and use them for navigation. The wireless sensors serve as tokens and do not communicate with each other. In [8] [13] mobile sensors are used to heal coverage holes in a sensor network.

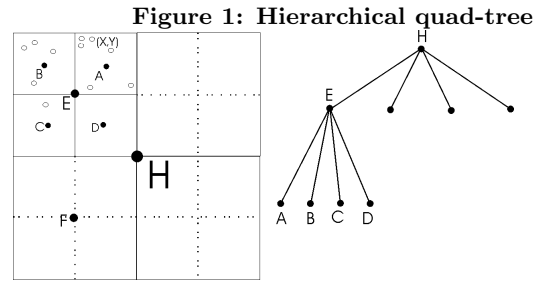
Data collection is conceptually more complex than routing. It involves not only routing data to a certain destination, but also such issues as data compression, data aggregation, filtering according to certain criteria defined by queries. Routing decisions can be influenced not only by destination address and QoS parameters, but also on the content of the data being routed (data centric routing).

However, because network routing provides a pure algorithmic routing without dealing with other complexities, it would be useful to investigate how delivering data to a mobile sink can be solved using existing routing protocols. The ad hoc routing protocols can be divided into reactive and proactive. Proactive protocols such as RIP always maintain route information and can provide a route to any destination immediately. The drawback of proactive protocols is the amount of overhead they create to maintain route information in the presence of dynamic network topology and intermittent links. Reactive protocols (AODV, DSR, and OLSR) create route information on demand to an incoming query. The advantage is that route is discovered on demand to an incoming query and can be cached for future requests. However, each route discovery creates a broadcast flooding throughout entire network which is very expensive.

It is easy to see that proactive or reactive protocols would perform poorly for routing data to a mobile sink. Proactive protocols would always broadcast link availability whenever a mobile user comes into contact with a sensor creating overhead. Reactive protocols would require flooding entire network with a route discovery request whenever a sensor needs to send data to a mobile sink. The problem here is that our scenario is very specific: the network is neither mobile nor static. The sensors are static and the sink is mobile. We think that ideas from location tracking systems could be useful to solve the problem of data delivery to mobile sinks. A location tracking system would track the position of a mobile sink and would provide this information to any sensor which needs to send data to a mobile sink.

### 3. INITIAL RESEARCH RESULTS

As a result of the research we have designed a distributed spatial index for sensor networks that adapts to the query and data update rates. The index is based on the hierarchical quad trees and uses a combination of proactive and reactive approaches to query individual sensors. The details of the protocol are available in [6]. Below is a short description of a scenario where it could be used as well as a novel



approach that we take.

Let us consider the scenario on large scale environmental monitoring. Thousands of sensors are distributed over large scale area and collect air pollution levels. The sensors are located in hard to reach places and have to run on a single battery for extended periods of time. So any solution has to be energy efficient. The solution also has to be scalable, because the network can contain thousands of nodes.

The network has to serve continuous queries from various users looking for specific data. The typical query would be a query like: What is the location of all abnormal air pollution levels within range  $R$  from the point  $(X,Y)$ . One way to answer this question is to flood the network with the query and wait for replies from each sensor. But clearly this is not energy efficient as it requires lots of communication overhead, especially in case of continuous queries. It is clear we need some kind of index to collect this information from sensors and provide it to users who run the queries. So our goal is to build an index of all data, which is scalable and energy efficient

Our distributed index uses two novel approaches: a static clustering algorithm and a combination of proactive and reactive modes for index updates.

We need to split the network into clusters and assign a cluster head which would be responsible for indexing data inside the cluster. We do this by decomposing an entire area into a set of disjoint hierarchical square cells where each cell consists of four smaller cells and so on (1). A cell at each level of hierarchy has a cluster head responsible for indexing data in that cell. Cluster heads at the lowest level of hierarchy collect information directly from sensors. Each cluster head is located inside its cell at a fixed relative location. In the example given in 1 the cluster head is located in the centre of each cell. This location in the picture (centre of the cell) was chosen quite arbitrarily, but it has to be uniform throughout all cells. Since all the cluster heads are located at predetermined coordinates there is no need for distributed cluster selection algorithms or advertising cluster head location. This allows us to create a distributed indexing hierarchy without any communication overhead. Given any pair of coordinates  $(X,Y)$ , the location of all cluster heads responsible for this area, at any level of the hierarchy, is easy to calculate. In the example given in 1, the cluster heads responsible for region  $(X,Y)$  are A, E, H, respectively.

We now show how the index tree is created and how it

adapts to query and local event rates. Maintaining an index involves a certain amount of communication overhead as sensors keep sending all event information periodically to cluster heads. To minimize this overhead the index tree is created on demand; that is, it is created and maintained only for those parts of the network that do receive queries. The tree is also adaptive in the way it resolves incoming queries. Each branch of the tree can be in on the following three states: i) proactive ii) reactive iii) pruned. Proactive branches periodically send information about all the events to their parent. The parent cluster head then always has up-to-date information about the availability of data and can use that information to answer queries. Reactive branches do not send updates to their parents; however, they periodically send keep-alive notifications which piggyback binary information about the availability of each data type in that region. Pruned branches are not active because they have never been used or there has been no recent activity on them. The difference between reactive and pruned modes is that, in reactive mode, cluster heads still send binary information about each event type to their parent. In pruned mode the cluster heads do not send any information to their parent, although they can still serve local queries. The state of each branch depends on the rate of event updates and lookup queries for data in the relevant cell. In general, our approach uses reactive mode requests for branches with high data update rates, proactive mode for children with less frequent update rates and it prunes inactive branches.

As an example, let us consider how a query is formed and resolved. A requester sends the query to a cluster head responsible for the minimum bounding square covering the area of interest. As was noted earlier, the location of all cluster heads is predefined, so it should be easy to calculate. The query should contain information about an area of interest (in the form of a polygon), the type of events it is interested in and the maximum tolerated latency. Upon receiving a query a cluster head creates a tree, if this does not exist, and forwards the query down all the branches covering the spatial area of interest. If the tree does already exist, it checks the mode of each of its branches before forwarding the query. In case of proactive branches, there is no need to forward the query, as all information should be already available. The cluster head has to forward the query to reactive and pruned branches. For the reactive branches it will check if the branch contains the given data type before forwarding the query. As we have seen from the example, the tree branches can have different operating modes, depending on the rate of queries and event updates. The cluster head constantly evaluates those rates and switches to the mode which imposes the smallest communication overhead.

As a result of initial research we have designed a distributed index that adapts to local event and lookup query rates to minimize the amount of communication overhead. There are still more interesting questions on how to improve our solution by introducing data aggregation techniques and other techniques.

#### 4. ACKNOWLEDGEMENTS

I would like to thank my advisor Cecilia Mascolo for guidance and helpful suggestions. I am also grateful to EPSRC and Vodafone for supporting my research.

#### 5. REFERENCES

- [1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):pp. 102–114, 2002.
- [2] M. Batalin, G. Sukhatme, and M. Hattig. Mobile robot navigation using a sensor network. In *In IEEE International Conference on Robotics and Automation*, pages 636–642, New Orleans, Louisiana, Apr 2004.
- [3] S. Bhattacharya, H. Kim, S. Prabh, and T. Abdelzaher. Energy-conserving data placement and asynchronous multicast in wireless sensor networks. In *Proc. of the 1st international conference on Mobile systems, applications and services*, pages 173 – 185, 2003.
- [4] L. Capra. Mobile computing middleware for context-aware applications. In *Proc. of the 24th International Conference of Software Engineering (ICSE 2002), Doctoral Symposium*, pages 723–724. ACM Press, May 2002.
- [5] A. Chakrabarti. Exploiting predictable observer mobility for power efficient sensor network communication. 2003.
- [6] V. Dyo and C. Mascolo. Adaptive distributed indexing for spatial queries in sensor networks. In *Proc. of the 8th Int'l workshop on Mobile Databases and Distributed Systems (MDDS'05)*, pages pp. 1103–1107, August 2005.
- [7] W. Emmerich. *Engineering Distributed Objects*. John Wiley & Sons, Chichester, UK, 2000.
- [8] S. Ganeriwal, A. Kansal, and M. Srivastava. Self-aware actuation for fault repair in sensor networks. May 2004.
- [9] B. Greenstein, D. Estrin, R. Govindan, S. Ratnasamy, and S. Shenker. Difs: A distributed index for features in sensor networks. In *First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA 2003)*, 2003.
- [10] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proc. of the 6th annual international conference on Mobile computing and networking*, pages 56 – 67, 2000.
- [11] B. Karp and H. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *Proc. of the 6th annual international conference on Mobile computing and networking*, pages 243 – 254, New York, NY, USA, 2000. ACM Press.
- [12] A. LaMarca, D. Koizumi, M. Lease, S. Sigurdsson, and G. Borriello. Making sensor networks practical with robots. LNCS 2414, pp. 152-166, 2002.
- [13] G. Wang, G. Cao, and T. Porta. A bidding protocol for deploying mobile sensors. Nov 2003.