A Framework for Querying Sensor Networks Using Mobile Devices[1]

Shourui Tian†, Sol M. Shatz† and Yang Yu‡

†Department of Computer Science
University of Illinois at Chicago
Chicago, IL 60644 USA
stian,shatz@cs.uic.edu

‡Pervasive Platforms and Architecture Lab
Application Research Center, Motorola Labs
Schaumburg, IL 60196 USA
yang@motorola.com

***Abstract*—an interplay between mobile devices and static sensor nodes is envisioned in the near future. This will enable a heterogeneous design space that can offset the stringent resource and power constraints encountered in traditional static sensor networks by taking advantage of the more powerful mobile devices. As such, we present a systematic framework for end-to-end query processing, using a two-layer architecture that consists of mobile devices at the upper layer and static sensor nodes at the bottom layer. One of our key goals is to achieve energy-efficient query injection and data collection by leveraging the mobility and transmission flexibility of objects at the upper layer. We propose a pull query model that contains staged operations including query generation, query routing, query injection, and query result routing. In the context of this model, we investigate a suite of techniques for the scenario with location-ignorant sensor nodes.**

***Index Terms*—mobile objects, query processing, sensor networks.**

I. INTRODUCTION

Stringent power constraints and limited computation and communication capabilities are key issues in the development of sensor networks. For example, a Berkeley Mote powered by two AA batteries can operate for about one year in the idle state, but only one week when fully loaded. In addition, large-scale sensor network applications impose a demand for cheap, small, low-power sensor nodes, making it impractical to equip sensor nodes with GPS-receivers. While various localization techniques are evolving [1], localization for very large-scale sensor networks is still in the research stage, especially when there is the need to obtain accurate location information under restrained energy conditions. Thus, we focus on location-ignorant sensor networks, which impose challenges on various aspects for query processing in the network, including both query routing and results gathering.

In contrast to sensor networks, mobile wireless networks have much relaxed power constraints. Various communication technologies, including Cellular, Wi-Fi, WiMAX, and Bluetooth, have been developed for connecting billions of electronic products, such as PDAs, cell phones, laptops, and cars. Moreover, GPS-receivers are reasonably affordable on such mobile devices.

We observe that a heterogeneous design space consisting of both static sensor nodes and mobile devices can successfully offset the stringent resource and power constraints in traditional sensor networks. Such an interplay between mobile networks and sensor networks also links the mobile devices as query requesters and data consumers directly to sensor networks that are responsible for sensing the physical world. We propose a framework for query processing that is based on the "PULL" query model [2] in a two-layer network structure, including a mobile network at the upper layer and a wireless sensor network at the bottom layer. Hereafter, the term "mobile objects" refers to mobile devices operated by users. One novel aspect of our approach is that mobile objects can take advantage of each other's independent motion plans to do a form of opportunistic query processing. In our framework, mobile objects take on four roles: query generator, query carrier, query injector and query result collector. By assigning these roles to mobile devices, we expect to simplify the operations of sensor nodes, thus shifting the energy burden from sensors to mobile objects.

The presented framework is designed for time-tolerant, end-to-end query processing, which consists of four key phases: query generation, query routing, query injection, and query result routing. Specifically, after being *generated* by a querying mobile object M_s, a query is first *routed* in the mobile network layer in an effort to get the query "close" to the target query region. Geographic routing can be used to serve this purpose. When a mobile object that is "close enough" to the query region receives the query, it analyzes the query to determine *if* and *where* the query shall be injected into the sensor network layer, based on the mobile object's current location and velocity, and the query's expiration time. A mobile object *injects* the query into the sensor network when the mobile object arrives at an expected "injection point" location. We denote the injecting mobile object as M_i. Upon receiving the query, sensor nodes perform the required sensing task, and then attempt to *route* the query results back to M_i, typically by disseminating the results to some "intermediate" sensor nodes. When M_i receives the results, it routes the results back to the querying object M_s, again via geographic routing. We also compensate for the mobility of M_s by performing a controlled flooding when the results are

[1] This material is based upon work supported by the U.S. Army Research Office under grant number W911NF-05-1-0573.

routed to the proximity of M_s's original position, i.e. its position at the time of initial query routing

An implication of our framework is that by exploiting the wide-area mobility of mobile objects, our research has the potential to integrate far-apart sensor networks and provide a framework for future pervasive computing environments.

A. Related Work

To conserve sensor node power and simplify routing, base stations in sensor networks are typically assumed to be fixed [3]. The MetroSense project [4] provides a three-tier architecture: a server tier; a sensor access point tier, and a sensor tier consisting of mobile and static sensors. One of the key aspects of MetroSense is to virtually extend the sensing range of a static node by delegating its sensing task to a passing by mobile sensors. In some recent research, mobility of base stations has also been exploited. The TTDD scheme [5] provides a Two-Tier Data Dissemination approach to reduce battery consumption and transmission collision during frequent location updates from multiple sinks to sensor nodes. Another data dissemination protocol, SEAD [6], is proposed to minimize energy consumption in building the dissemination tree and in disseminating data to mobile base stations. In addition, the approach in [7] focuses on the topology control process for mobile base stations and application nodes, which serves as a type of "super" sensor node – it receives raw data from sensor nodes, creates a comprehensive local-view, and forwards the composite bit-stream toward a base station. The Data MULEs approach [8] presents a three-tier architecture and "MULEs" pick up data from sensors in close transmission range, buffer the data, and then drop off the data to wired access points. However, as the designers of this approach noted, the energy consumed during radio monitoring can be very high because each sensor must continuously listen in order to identify a passing-by MULE. The Tenet approach [9] implements a two-tier network: a lower tier consisting of motes and an upper-tier containing relatively less resource-constrained masters.

In general, the research ideas mentioned above, except MetroSense, are based on an event-driven "PUSH" query model. We adopt an alternative scheme, the "PULL" query model. In this scheme, queries are generated by mobile objects, and then disseminated into the sensor network through other mobile objects. The results pulled out from the sensor network are then routed back to the querying objects via the mobile network layer.

Moreover, previous works assume that users and mobile base stations play two distinct roles in sensor networks, with base stations serving as simple information "senders" or "collectors" for users on an existing network. In contrast, our approach unifies the roles of mobile users and mobile base stations. We extend their role to behave as query carrier and result collector. This requires cooperation among mobile objects, which are connected via ad hoc wireless networks.

Although the MetroSense project and our framework share the idea of exploiting mobile nodes that operate in a sensor environment, there are several fundamental differences. MetroSense and our approach assign opposite roles to static and mobile nodes. In MetroSense, mobile sensors interact with a static communication infrastructure, while our approach emphasizes the integration of mobile devices with static sensor networks. Our approach focuses on conserving sensor node power during query processing by limiting the communication burden of sensor nodes; MetroSense focuses on accomplishing a sensor's predefined sensing task when the sensor's sensing range cannot reach a desired sensing region. Finally, our query-oriented research is based on a two-layer architecture with mobile objects serving as source, carrier, injector and collector of queries, whereas MetroSense uses a three-tier architecture that includes a server layer.

Our approach shares some similar ideas with Tenet [9], e.g., both taking advantage of the mobile objects or masters in the upper tier to offset the constraints in the lower layer. But, Tenet prohibits communication between motes, aiming to increase system manageability and reduce complexity.

The work in this paper extends the results provided in [10] by defining the overall 2-layer system architecture and presenting techniques for each of the query processing phases. In [10], the focus was only on the issue of query injection.

B. Paper Organization

In Section 2, we describe the target applications and key properties of mobile objects and sensor nodes. Our system architecture is presented in Section 3. In Section 4, we discuss details on four key query-processing phases, as well as associated techniques. We conclude our paper in Section 5.

II. Target Applications and properties

A. Applications

The type of two-layer networks we consider in this paper can be applied to a wide array of applications when high density mobile objects exist, such as in urban environments.

Thousands of sensor nodes can be densely scattered over some area for monitoring environmental information and aiding mobile users, who might have mobile devices connecting each other through peer-to-peer network. For example, sensors may be densely dispersed along a highway to help monitor traffic and road conditions. Consider an advanced automotive agent system, such as the WILLWARN system [11] sponsored by European automakers, that might serve as an intelligent driving assistant. In the course of its route planning activity, this agent may be interested in road conditions at some remote region (a region not within the vehicle's mobile device local transmission range). In this case the (mobile) agent can transmit a query to other vehicle-based agent systems, which in turn route the query for eventual injection to the target region. Therefore, the agent system can obtain useful information such as icy road conditions or traffic jams in advance, and plan some alternative route. Note that travel plans of individual mobile objects can be independent of the queries that an object happens to be carrying.

B. Mobile Objects

We consider mobile objects with relatively powerful computation and communication capability. We assume mobile objects are supported by a rechargeable battery, and equipped with localization devices that also provide a global clock. We also assume that mobile objects have two radios operating in different frequency bands [12]. For example, one radio may operate in the 915 MHz band to communicate with sensor nodes

while the other radio operates in the 2.4 GHZ band to communicate with other mobile objects. In addition, a mobile object is assumed to be able to adjust its transmission power when communicating with sensor nodes [13], so as to tune the number of sensor nodes that must be actively engaged during query injection.

C. Sensor Nodes

To facilitate the design of our system architecture and query processing mechanisms, we assume a large number of static sensor nodes to be deployed with high node density, each with limited battery power and computational capability. Also, for our purpose, we assume that sensor nodes are not aware of their location information since localization techniques are still in the research stage, especially those techniques that can reduce energy cost and improve accuracy for very large-scale networks. However, as localization techniques mature, it may be possible to take advantage of new localization methods (e.g., [1][14]) for large-scale, low-cost sensor networks. The presented framework is extensible to cover cases where location information is available to sensor nodes.

III. System Overview

A. Query Details

For the purpose of this paper, we consider delay-tolerant queries, i.e., the querying mobile object requires the results to be delivered in a relatively relaxed time frame, ranging from seconds to minutes. For simplicity, we assume a circular target region, referred to as the *query region* (*QR*). For example, a mobile object MO100 at location (10, 100) may issue a query at time 500: "report the temperature associated with the query region centered at location (45, 90), with a radius of 10 meters, before time unit 530."

A *query q* is a 4-tuple: *q* = (*q_id, object_info, q_region, and q_expiration*), where

q_id is a unique query identifier;

object_info = (*object_id*, *object_location*, *object_velocity, q_time*) is a container for information regarding the mobile object generating the query *q*;

object_id is a unique identifier of the querying mobile object, *object_location* and *object_velocity* are the (x, y) coordinate position and the velocity of the querying object at the time it initiated routing of the query and *q_time* is the time stamp of the query.

q_region = (*S, r)* is the query region centered at *S* = (x, y), with radius *r*;

q_expiration is the expiration time of the query.

In terms of the above example, assuming the object is moving at a speed of 20 m/sec, the query can be expressed as the 4-tuple: *q* = (*q1,* (*MO100,* (*10, 100*), *20, 500*), ((*45, 90*), *10*), *530*), where *q1* denotes a query for temperature.

We assume that each mobile object can store either locally-generated queries or routed queries in a local database. Considering the advances in low-cost, high capacity memory, we assume a sufficiently large local storage space, thus not concerning ourselves with storage constraints at this time.

B. System Architecture

Based on the properties of mobile objects, sensor nodes and queries, we propose a new two-layer network system consisting of a mobile network layer and a sensor network layer. The basic architecture is depicted in Figure 1. Although mobile objects and sensor nodes are physically located in the same two-dimensional coordinate system, we logically classify mobile objects as being in the "upper" layer, and sensor nodes as being in the "lower" layer, because of their different properties.

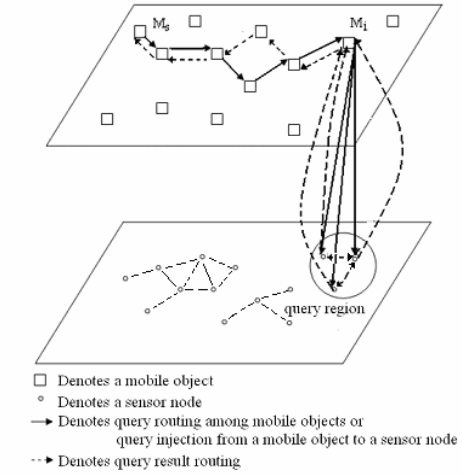

Figure 1. System architecture

Although there may be multiple queries in progress at any time, to simplify our discussion we consider only the processing of a single query since each query is handled independently. Our query processing consists of four sequential phases as illustrated in Figure 2: query generation, query routing, query injection, and query result routing.

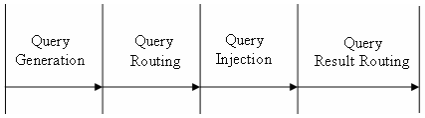

Figure 2. End-to-end query processing

Since sensor nodes are location-ignorant, they can only support data routing by "blind" message flooding. Moreover, sensors do not know if they are located within the query region specified by a query. Thus, one of the key challenges for the mobile object that finally injects a query into the sensor network is to choose the best location to inject the query (the injection point). Moreover, the lack of localization information requires an energy-efficient method to propagate query results from sensor nodes to a mobile object. We will discuss our techniques to handle query injection and communication control between sensor nodes in Section 4.

One final comment about our approach is the potentially large conservation of sensor node power due to the fact that a sensor node only reacts when it has received a query from some mobile objects. Thus, sensor nodes can stay in a low-power mode during most of their life, using a radio-triggered wake-up scheme [15].

IV. QUERY PROCESSING PHASES

A. Query Generation

A mobile object may generate queries either based on spontaneous user interest, or in a planned fashion. For the former case, for instance, a driver may request traffic information for a particular road segment. For the second case, an intelligent agent-based automotive system, as alluded to earlier, may automatically query road conditions within the next few miles. Once a query is generated by a mobile object, it is stored locally in preparation for query routing.

B. Query Routing

After a query is generated by a source mobile object M_s, the query is routed to other mobile objects, instead of being distributed to sensor nodes immediately as in traditional sensor networks. Mobile objects can route the query among themselves via various network connections, including cellular, satellite, or WiFi. We observe that using existing cellular or satellite systems would require extra coordination (e.g., knowing the phone number of a mobile object at the query region) and infrastructure support from carriers (e.g., semantic processing of short messages). Thus, we prefer an infrastructure-free peer-to-peer mechanism, including the ad hoc mode of 802.11 and various DSRC [16] compliant technologies for vehicular ad hoc networks (VANETs), to route the queries. Since we consider a region-based query and assume knowledge of location information for mobile objects, geographic routing [17] is a natural choice. So, a mobile object always selects a neighbor closer to the destination as the next forwarding hop, so that the query makes progress toward the query region. Existing techniques to handle a potential "dead end" can be adopted [18].

A mobile object can store and route queries either locally generated or received from neighbors. For received queries, the mobile object checks the query's validity by comparing its local time to the *q-expiration* in the query record. Expired queries are dropped. Also, to overcome the unreliability of wireless communication and object mobility, a routed query may be re-transmitted if no response is received after a pre-defined timeout.

C. Query Injection

If a mobile object has no neighbor object closer to the query region, for the purpose of further query routing in the mobile network layer, the mobile object becomes a potential candidate for injecting the query to the sensor network. Multiple query injection candidates can exit for one particular query. Here we describe some key aspects of query injection; further details can be found in [10].

To avoid blind, energy inefficient flooding by sensor nodes, a mobile object, M_i, should inject a query when the query region is (partially) within the object's transmission range. The region that is reached by the query is referred to as the *dissemination region* (*DR*) of the query. Based on a mobile object's current velocity, the object's short-term trajectory can be predicted in real-time. Figure 3 shows an injectable-query segment $\overline{C'E'}$, which defines the locations during which the distance between the mobile object and the center $S(x, y)$ is less than or equal to $(R+ r)$ (the sum of the transmission range of the object and the query region radius).[2]

A mobile object can inject a query to the sensor network when it reaches an *injection point* (*IP*), which can potentially be any point on $\overline{C'E'}$. This allows the mobile object to choose a desired injection point in order to make a trade-off between coverage of the target query region and energy cost associated with the sensor nodes involved in the query.

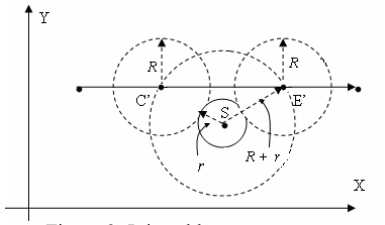

Figure 3. Injectable-query segment

Sensor nodes within the entire dissemination region—even those not within the query region—are awakened to perform the sensing task. To conserve energy while ensuring a reasonable query quality, it is crucial to maximize awakened nodes that are within the query region. Intuitively, this requires query injection at a location along M_i's trajectory that is closest to the center of the query region.

A metric called the *query region coverage rate* (*QRCR)* is introduced to measure the effectiveness of query injection.

$$QRCR = \frac{\#\ of\ awakened\ sensors\ in\ query\ region}{\#\ of\ sensors\ in\ query\ region}$$

We evaluate the relationship between *QRCR* and the injection point of the mobile object, when the dissemination region and query region are partially covered. We conclude that *QRCR* is a decreasing function with respect to the distance between injection point and the query region center *M*, assuming a large number of sensors uniformly distributed in the environment. Detailed formulation and proof of QRCR are available in [10].

Figure 4 shows the numerical analysis result of *QRCR*, given R=500, r=400 and M_{min}=500. *QRCR* is maximized when the mobile object injects a query at the *optimal injection point* M= M_{min} =500.

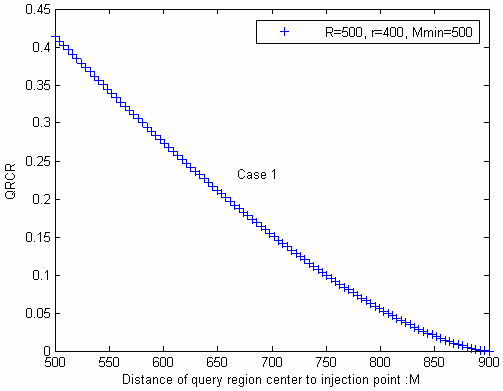

Figure 4. Numerical simulation of *QRCR*

[2] Although we acknowledge irregular and dynamic communication range for realistic radio models, we adopt a circular model for tractable analysis, which is quite common in the literature, e.g., [19].

But, maximizing *QRCR* may still result in query reception by some number (possibly many) sensor nodes that are *not* within the query region. In Figure 5, which represents a case when the coverage rate is optimal (equal to 1), we see many such sensor nodes that are being "reached" inadvertently. We refer to these sensor nodes as "unintended sensor nodes." Those sensor nodes outside the dissemination region are called "uncovered sensor nodes" (they are not covered/reached by an injected query).

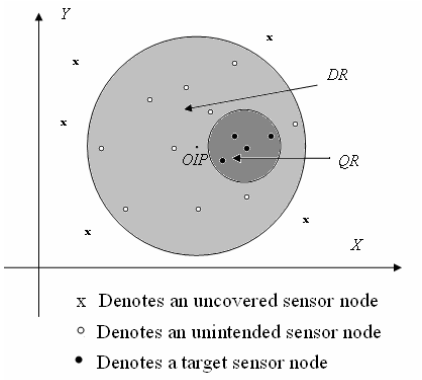

Figure 5. Optimal QRCR, but "unintended" sensor nodes

To rectify this situation, M_i can adjust its transmission range to tune the fraction of awakened "target sensor nodes" and reduce awakened "unintended sensor nodes". This can be measured by a coverage metric, *dissemination region coverage rate* (*DRCR)*:

$$DRCR = \frac{\#of\ awakened\ sensors\ \ in\ query\ region}{\#of\ sensors\ \ in\ \ dissemination\ region}$$

Assuming a large number of uniformly distributed sensors with a given node density, Figure 6 demonstrates numerical results for *DRCR* when a mobile object injects a query to a target query region of radius *r*, with varying distance, *M*, from the center of the query region and with adjustable transmission range, *R*.

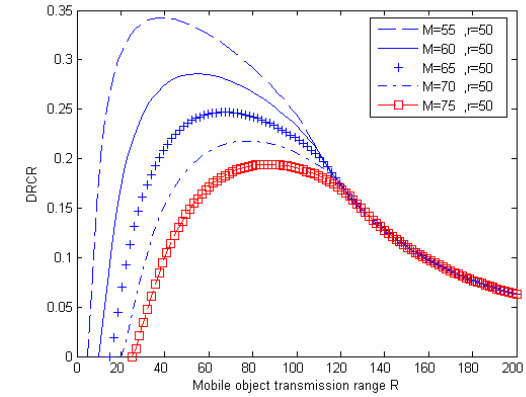

Figure 6. Impact of transmission range on *DRCR*

We observe that the *DRCR* improves when *R* decreases to a certain threshold, after which *DRCR* then decreases with *R*. This is expected, since before reaching the threshold, the decrease in R effectively reduces the number of awakened nodes outside the query region. However, below the threshold of *R*, more nodes within the query region are not covered, which reduces *DRCR*. The values of R at the peaks of the curves provide the optimal transmission range for query injection under various circumstances.

Figure 6 shows the case when the dissemination region and query region are partially overlapped. Complete case studies are available in [10].

Before injecting a query, if M_i changes its velocity, including its motion speed or direction, M_i will recalculate its optimal injection point and optimal transmission range.

D. Query Result Routing

Upon receiving a query, sensor nodes perform the required sensing task. A key operation is then to efficiently route the sensing results back to a mobile object. Since the transmission range of sensor nodes is typically much smaller than that of a mobile node, sensor nodes that cannot transmit far enough to reach a mobile object must transmit their query results to some intermediate sensor nodes, which are able to send the results to a nearby mobile object. To prevent network-wide flooding, a *Return Hop Counter (RHC)* can be used to implement a controlled flooding. The *RCH* is pre-calculated by a mobile object during the query injection phase, and attached to the injected query. In the *RHC* formula, c is a small constant to counter the effect of zigzag paths that might exist from sensor nodes to M_i.

$$RHC = \frac{Mobile\ Object's\ Transmission\ Range}{Sensor\ Node's\ Transmission\ Range} + c$$

1) Routing at the Sensor Network Layer

After performing the required sensing task, sensor nodes broadcast *query result* packets *RP* = (*q_id*, *object_info*, *q_expiration*, *RHC*, *sensor_id, q_data*), where *sensor_id* is the unique id for a sensor node and *q_data* is the sensor reading (after proper data fusion or processing). In dense sensor networks, *RHC* is used as an upper bound on the number of hops that an *RP* is re-transmitted by sensor nodes (See Figure 7). Every re-transmission decrements *RHC* until *RHC* reaches zero, at which time the RP is dropped.

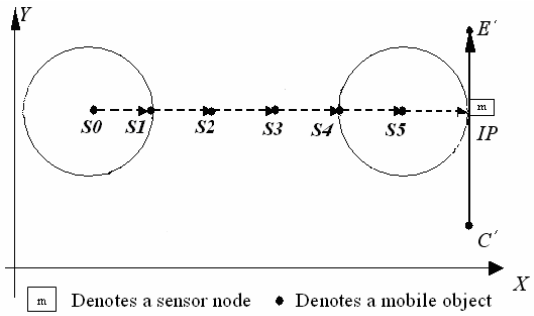

Figure 7. An example of query result routing with RHC=6

Also, sensor nodes temporarily cache the (*q-id*, *sensor-id*) pair of received *RP*s to identify duplicated packets. Moreover, various optimization techniques for controlled flooding can be used in this context for efficient dissemination of the result packets without incurring overwhelming traffic.

2) Routing From Sensor Node to Mobile Object

Since we embed *object_info*, which contains *object_location*, in the query result packets, we do not require that result packets be transmitted to the injecting mobile object. In fact, any mobile object that receives a query result packet *RP* may start routing the packet to the source querying object that initiated the query. However, the aforementioned *RHC* counter allows the query results to reach at least the injecting object with high probability, if the packet transmission speed and the sensor's query –processing speed are much faster than the movement speed of mobile objects – as we would expect.

3) Routing at the Mobile Layer

After receiving query results, a mobile object again uses a geographic routing scheme to route query results back to the querying mobile object. Since multiple mobile objects may receive the same *RP*, mobile objects can cache recently routed packets to identify and drop duplicate packets.

A key challenge is that when query results are routed back to *object_location*, the querying mobile object may have since moved to a new location. To cope with this problem, we propose a controlled flooding of *RP*s once they reach *object_location* but have failed to reach the querying mobile object. The flooding is performed within a circular region centered at *object_location*, with radius $|local\_time - q\_time| * |object\_velocity|$. In the worst case strategy, |*object_velocity*| can be set to the highest magnitude speed based on a history of recent object speeds. Thus the query result is flooded within a region whose size is based on the predicted distance position of the source mobile object.

This controlled flooding at the last hop is expected to eventually route the query result to the querying mobile object, M_s. If M_s does not receive any query result before *q_expiration*, the query request fails – this might occur because (1) no sensor node covers the query region, (2) no mobile object is able to route the query close enough to the query region, or (3) no route can be established to route the results back to the querying object. We are investigating mechanisms to identify the above failure cases and corresponding approaches to handle them.

V. Conclusion and Future Research

We presented a query processing architecture for networks composed of mobile objects operating within the context of a sensor rich environment. Key properties of mobile objects are used to offset the constraints associated with sensor nodes.

One limitation that is inherent to our approach is the potential for query failures, i.e., the inability to obtain a query result within the specified time and location constraints. For example, with insufficient density or undesired distribution of mobile objects, it may be difficult or even infeasible to inject queries to a specified target query region. Future research includes justification of our approach with traditional sensor networks with fixed infrastructure, and extensions to efficiently resolve identical or similar queries from multiple mobile objects.

ACKNOWLEDGMENT

We wish to thank the reviewers for very useful comments that improved the ideas of this paper.

References

[1] R. Stoleru, T. He, J. Stankovic and D. Luebke, "High-Accuracy, Low-Cost Localization System for Wireless Sensor Networks," *The Third ACM Conference on Embedded Networked Sensor Systems*, San Diego, USA, Nov. 2005.

[2] A. Trigoni, Y. Yao, A. Demers, J. Gehrke and R. Rajaraman, "Hybrid Push-Pull Query Processing for Sensor Networks," *Proceedings of the Global Internet Workshop on Sensor Networks* (WSN), Karlsruhe, Germany, Feb. 2004.

[3] Y. Yu, B. Krishnamachari, and V. K. Prasanna, "Energy Minimization for Real-Time Data Gathering in Wireless Sensor Networks," *IEEE Trans. on Wireless Communication*, Vol. 5, No. 11, pp. 3087-3096, Nov. 2006.

[4] S. B. Eisenman, N. D. Lane, E. Miluzzo, R. A. Peterson, G. Ahn, and A. T. Campbell, "MetroSense Project: People-Centric Sensing at Scale," *The Workshop on World-Sensor-Web (WSW 2006)*, Boulder, Oct. 2006.

[5] H. Luo, F. Ye, J. Cheng, S. Lu and L. Zhang, "TTDD: Two-layer Data Dissemination in Large-scale Wireless Sensor Networks," *ACM Wireless Networks*, Vol. 11(1-2), pp. 161-175, Jan. 2005.

[6] H. S. Kim, T. F. Abdelzaher and W. H. Kwon, "Minimum-Energy Asynchronous Dissemination to Mobile Sinks in Wireless Sensor Networks," *The First ACM Conference on Embedded Networked Sensor Systems,* Los Angeles, Nov. 2003.

[7] J. Pan,, L. Cai, Y. Hou, Y. Shi and X. Shen, "Optimal Base-Station Locations in Two-Layered Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, Vol. 4(5), pp. 458-473, 2005.

[8] R. Shah, S. Roy, S. Jain and W. Brunette, "Data MULEs: Modeling a Three-Tier Architecture for Sparse Sensor Networks," *The First IEEE International Workshop on Sensor Network Protocols and Applications*, Alaska, USA, May 2003.

[9] O. Gnawali, B. Greenstein, K. Jang, A. Joki, J. Paek, M. Vieira, D. Estrin, R. Govindan and E. Kohler, "The TENET Architecture for Tiered Sensor Networks," *The* 4th *ACM Conference on Embedded Networked Sensor Systems,* Boulder, CO., Nov. 2006.

[10] S. Tian and S. M. Shatz, "Optimizing Query Injection from Mobile Objects to Sensor Networks," *The 8th International Symposium on Autonomous Decentralized Systems*, Arizona, USA, Mar. 2007.

[11] WILLWARN: http://prevent-ip.org/en/prevent_subprojects/safe_speed_and_safe_following/willwarn/

[12] V. Kottapalli, A. Kiremidjian, J. Lynch, E. Carryer, T. Kenny, K. Law and Y. Lei, "Two-Tiered Wireless Sensor Network Architecture for Structural Health Monitoring," *The 10th Annual International Symposium on Smart Structures and Materials*, San Diego, USA, March, 2003.

[13] J. Gomez and A. T. Campbell, "A Case for Variable-Range Transmission Power Control in Wireless Multihop Networks," *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[14] T. He, C. Huang, B. M. Blum, J. A. Stankovic and T. F. Abdelzaher, "Range-Free Localization and its Impact on Large Scale Sensor Networks," *ACM Transactions on Embedded Computing System*, Vol. 4, Issue 4, 2005.

[15] L. Gu and J. Stankovic, "Radio-Triggered Wake-Up Capability for Sensor Networks," *The 11th IEEE Real-Time and Embedded Technology and Applications Symposium,* Toronto, May 2004.

[16] DSRC, http://grouper.ieee.org/groups/scc32/dsrc/index.html.

[17] X. Li, G. Calinescu and P. Wan, "Distributed Construction of Planar Spanner and Routing for Ad Hoc Networks," *Proceedings of IEEE INFOCOM 2002*, New York, USA, Jun. 2002.

[18] H. Frey and I. Stojmenovic, "On Delivery Guarantees of Face and Combined Greedy-Face Routing Algorithms in Ad Hoc and Sensor Networks," *The 12th ACM International Conference on Mobile Computing and Networking*, Los Angeles, CA, Sept., 2006.

[19] R. Iyengar, K. Kar, and S. Banerjee, "Low-coordination Topologies for Redundancy in Sensor Networks," *ACM Mobihoc*, Urbana-Champaign, IL, May 2005.