Dynamic Resource Discovery for Wireless Sensor Networks

Sameer Tilak[1], Kenneth Chiu[1], Nael B. Abu-Ghazaleh[1], and Tony Fountain[2]

[1] Computer Science Department
SUNY Binghamton
[2] San Diego Supercomputer Center
University of California at San Diego

Abstract. As sensor networks mature and are increasingly deployed, they will play an integral role in everyday life as the interface between the physical and digital worlds. In contrast to the current generation of sensor networks that are application-specific and exposed only to a limited set of users, heterogeneous sensor networks will start to be used by users in their surrounding environment dynamically. The available sensors are likely to be dynamic (e.g., due to mobility) and heterogeneous in terms of their capabilities and software elements. They may provide different types of services and allow different configurability and access. A critical component in realizing such a vision is *dynamic resource discovery*. The nature of resource discovery requires modular standards that allow interoperation among heterogeneous software and hardware systems. In this paper, we develop a resource discovery protocol for sensor networks, outline some of the challenges involved, and explore solutions to some of the most important ones. Specifically, we first discuss the problem of what resources to track and at what granularity: in addition to the individual sensor capabilities, some resources and services are associated with sensor networks as a whole, or with regions within the network. We also consider the design of the resource discovery protocol, and the inherent tradeoff between interoperability and energy efficiency.

1 Introduction

Wireless Sensor Networks (WSNs) promise to revolutionize sensing in a wide range of civil, scientific, military and industrial applications. For example, they may be deployed around a city to monitor chemical and biological threats, in the wild to monitor ecological events in migration patterns [18], or to track a smoldering forest fire for conditions that might lead to an outbreak. The first generation of WSNs are primarily application-specific in terms of their infrastructure, capabilities, protocol/architecture and user set.

The success and increasing deployment of this first generation is driving towards a second generation that envisions commodity sensors and sensor networks providing standard services that can be composed by various clients into a wide range of applications. These sensor networks then form a critical, yet generic interface between the physical and digital worlds, converting physical qualities into measurements that can be then harnessed for a wide-ranging spectrum of applications. The impact of such a development would be profound: no longer would sensor networks be specialized networks serving a limited set of

This work was partially supported by NSF Awards SCI-0330568, CNS-0454298 and DBI-0446298, and Air Force grants FA8750-04-1-0211 and FA8750-05-1-0130.

users, but rather a basic infrastructure supporting and encouraging unanticipated functions and modalities of operation. Increasingly advanced and accessible applications will be enabled as sensor networks are shared dynamically and ubiquitously, allowing clients to unite disparate components on demand into *virtual sensor networks*. Some clients may even compose services spanning multiple sensor networks into a single end-to-end service for use by applications.

While many sensor networks are self configuring – sensors able to configure themselves to collaborate with other sensors in the network – they cannot generally interoperate with foreign sensor networks, nor provide their services to outside clients not familiar with the network's internal architecture. This limits current sensor networks to performing only the tasks for which they were specifically and painstakingly designed. Numerous sensor platforms and supporting software has been independently developed (Section 3 reviews some of these) making the problem of interoperating them challenging. Realizing this second generation vision thus requires the ability of clients to dynamically configure, discover, and access resources as they encounter unfamiliar sensor networks while moving through the environment. This ability will allow interoperating components to discover the available resources surrounding them and to configure themselves to interoperate with them. We motivate this model and describe the challenges in realizing it in Section 2.

A necessary precursor to interoperability is the ability to discover the resources with which interoperation is beneficial. It is also necessary to discover the parameters and attributes of these resources required for interoperation. Resource discovery in sensor networks is challenging for a number of reasons. Since different sensor networks are deployed and managed by different organizations, they generally use heterogeneous protocols architectures, security policies, management policies, and pricing mechanisms. Also, the nature of resources (sensing and coverage characteristics, connectivity characteristics, available energy, computational and storage capabilities, protocols and software elements) and services (e.g., localization, synchronization, calibration) are significantly different from those in traditional distributed systems. Furthermore, the potential mobility of sensors and clients introduce unique challenges [25]. Finally, the embedded nature of sensors place a premium on energy efficient solutions. The problem of resource discovery in the context of sensor networks thus requires significantly different solutions from traditional naming and resource discovery in distributed systems [2, 13]; we describe the relation to these and other works in Section 7.

The resource discovery problem in sensor networks can be broken down into two components: (1) Determining what resources to track and at what granularity to track them. In the context of sensor networks this is a challenging and multifaceted problem; and (2) determining the resolution protocol that queries the sensor network for the resource information and transfers this information to queries in an energy efficient way. Note that the two components are similar to other distributed resolution systems (e.g., DNS' servers and resolvers [22]). However, sensor network operation introduces many characteristics that make this problem unique such as resource attributes that are associated with the network as a whole or regions within, as well as energy efficiency concerns.

Resource discovery in sensor networks is discussed in detail in Section 4. In addition, this section presents an outline of the challenges involved in the two components of resource discovery. Section 5 presents a preliminary model for resource discovery that provides a scalable, distributed, and light-weight mechanism for resource discovery that accounts for the vagaries of wireless communication. Some implementation tradeoffs in the resource discovery protocol are evaluated in Section 6. Finally, Section 8 presents concluding remarks.

2 Motivation and Background

While this paper investigates resource discovery in sensor networks, the long term goal of our research is to enable ubiquitous inter-operation of sensor networks (for which resource discovery is required). In this model, users are readily able to discover and task sensor resources belonging to foreign sensor networks as well as compose complex tasks that cross multiple sensor networks. We view this ability as a natural extension of sensor networks towards ubiquitous operation. In this section we provide a motivating example for this vision, and describe the main challenges.

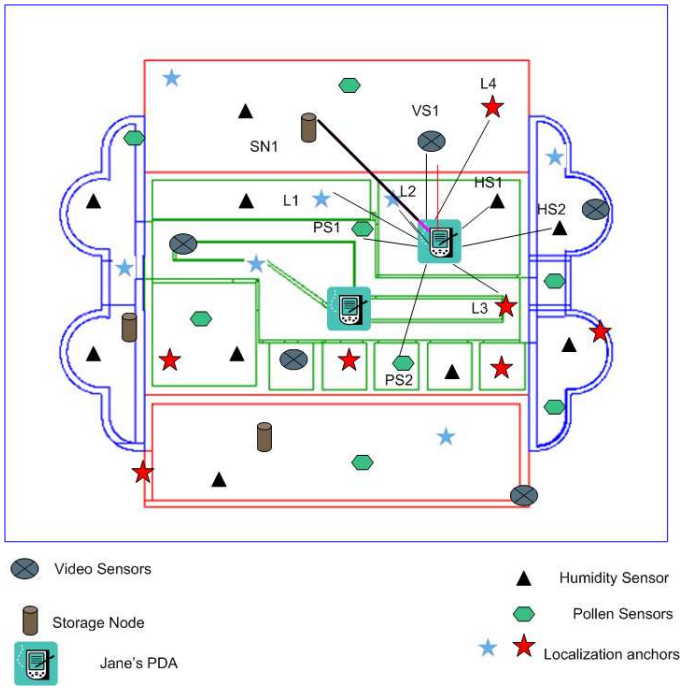

Fig. 1. Smart Mall

Consider the following scenario which describes how a future ubiquitous sensing infrastructure can revolutionize health monitoring and care. The year is 2055 and Jane, an elderly patient, is allergic to different airborne particles including pollen and dust. She also suffers from asthma and hypertension. In the past fifty years, various public and private entities, including a number of health care providers have instrumented Jane's town. She has subscribed to one health care provider. She carries a wireless access device with her that has stored her health profile, which includes information about her allergies, her medications, and emergency contacts.

When Jane enters a shopping mall, her device automatically discovers various sensors available in the neighborhood. An application on her PDA then uses her profile and finds out the available humidity and pollen sensors in the neighborhood. An application running

on her device automatically authenticates her using her subscription ID from her health care provider. In addition, it then tasks those humidity sensors ($HS1$,$HS2$) and pollen sensors ($PS1$,$PS2$) to send an alarm to her PDA when the humidity level and pollen grain level cross certain thresholds. As she walks through the mall, her device displays various areas as safe or unsafe based on humidity level.

Various sensors including blood pressure and temperature sensors are attached to her body and continuously monitor her vital signs. Upon detecting an abnormality in her breathing, a localization application on her access device immediately discovers nearby localization anchors ($L1$,$L2$,$L3$,$L4$), since, inside the mall, GPS localization is not available. Localization based on these anchors allows the device to calculate its location within a few inches. An emergency situation management application may report her current location to the emergency team.

Although recording detailed information of Jane's vital signs requires substantial storage space, it is crucial for emergency response and medical diagnosis. A logging application on her PDA discovers available storage nodes deployed recently by another health care provider. It then securely logs her on a storage node ($SN1$) and streams her vital signs onto those machines. The emergency team can now see her using a video camera VS_1 in the neighborhood and the professional help is given to her in a few minutes.

Jane was able to dynamically discover sensors and services spanning multiple networks and administrative domains. She was able to find out how to access them and even to task them. The entire process was completely automated and at every point her privacy was maintained and also secure mechanisms were in place for accessing the resources. It is easy to visualize other applications that can benefit from the ubiquitous presence of heterogeneous sensors that are able to interoperate to compose complex services.

The ability of sensor networks to dynamically discovery nearby resources and interoperate with them may be viewed as a first step towards a broader vision which we call Virtual Sensor Networks (VSN), drawing from the concept of virtual organizations [11]. In this model, the available heterogeneous sensing and computation resources are exposed to authorized users as a single virtual sensor network. Services may be discovered and composed across multiple sensors automatically and without user guidance to enable applications such as the one described in this section. Clearly, there are many technical, economic and personal privacy related challenges that must be overcome in order to realize this ambitious vision. These include: (1) Dynamic resource discovery across heterogeneous and diverse sensing platforms; (2) Dynamic service composition across multiple heterogeneous and independently administered networks; (3) Fair and QoS sensitive resource allocation among competing user interests; and (4) Security, privacy and economic settlement issues. VSN is a topic of our future work.

3 Sensor Hardware and Software Resources

Despite the relatively recent emergence of sensor networks as a field of study, already a large number of sensor hardware platforms and software elements (operating systems, networking protocols, data base systems, etc.) have emerged. There is a large variety in the capabilities and features of the proposed system. In this section, we briefly overview typical characteristics of the current generation of sensors.

Sensor hardware platforms vary in capability from miniature sensors such as the Berkeley motes [21], which are equipped with 8-bit microprocessors with a few kBytes of memory and low bandwidth low range radios, to PDA class sensors such as PASTAs [24]. Other sensor

platforms include Mantis [1], and WINS [26]; Table 1 presents a comparison of various sensor platforms.

Table 1. Sensor Platforms.

Platform	MICA	MicaZ	Mantis Nymph	RockWell WINS	Stargate
Processor	ATMega103L	ATMega128L	ATMega128L	Intel-StrongArm	Intel-Xscale
CPU speed	4 MHz, 8 bit	7.37 MHz, 8 bit	7.37 MHz, 8 bit	133 Mhz	400 MHz
RAM	4 kB	4 kB	4 kB + 64 kB	1MB	64 MB
Max. Data Rate	40(kb/s)	250(kb/s)	76.8 (kb/s)	100 (kb/s)	11(Mb/s)
Operating system	TinyOS	TinyOS	MOS	Win CE	Linux

On the software side, diversity exists at various layers, including operating system, programming languages, and communication protocols. For example, TinyOS, MANTIS OS (MOS), Linux, Windows CE, RTOS are all used as operating systems on the different platforms. At the MAC layer SMAC [40], TDMA [15], IEEE 802.11 and various other protocols exist. Directed diffusion [16], RAP [20], SPEED [14], and LEACH [15] represent some of the proposed routing and data gathering protocols. Due to various technical and economical reasons a sensor network deployed by an organization can consist of a diverse set of sensors. In fact, existing deployments such as Duck Island [7] capitalize on the heterogeneity of the sensors within the network. Sensor networks deployed by independent organizations might have heterogeneity even to a greater extent. This great diversity presents significant challenges to sensor network interoperability and motivates the need for resource discovery.

We believe that this heterogeneity is inevitable, and, furthermore, we believe that an attempt to impose uniformity on this diversity would stifle experimentation and innovation. But without some commonality, interoperability is not possible. We thus believe in the adoption of minimal, extensible standards that can promote interoperability by supporting the discovery of formats and protocols.

4 Dynamic Resource Discovery in Sensor Networks

An essential component in systems where foreign and heterogeneous elements interoperate is the ability to dynamically discover relevant information regarding resources and protocols. A client entering a new area generally has no knowledge of what sensor resources exist within it (in terms of hardware and software elements) or about the resources that these sensors track. Thus, properties such as protocols and formats that are relevant to interoperation must also be obtained. We call this problem *Dynamic Resource Discovery (DRD)*. In this section, we overview the challenges in DRD for sensor networks. The discussion is organized into the two major aspects of DRD: (1) Determining tracked attribute set; and (2) Resource Discovery Protocol.

4.1 Determining Tracked Attribute Set

The decision on what resources, services and other system attributes are tracked to support interoperability and service discovery is an important one. The candidate attribute set for

tracking may be classified along multiple axes. First, attributes can be classified into those that provide information regarding interoperability (e.g., protocols used or formats for messages), and those that describe the metastate of the system (resources or services provided). Standardization plays a role in determining what interoperability information to track (e.g., if sensors are standardized to be completely homogeneous, no interoperability information needs to be tracked). It is important to discriminate here between resource discovery and data collection. While both operations require collecting information from the sensors, DRD collects meta-information and not data. Although the resource discovery and data collection may be combined for efficiency in certain cases, it is likely that separating them will lead to a more effective and modular solution.

A different axis for classifying tracked attributes is the granularity of the resources. We classify attributes as simple (associated with single sensors) and complex (representing multiple sensors). Clearly, individual sensor resources are of interest. These include sensing resources (such as available sensors, their tolerances, and their coverage) as well as computational, storage and communication resources. Additional properties exist for the network as a whole (e.g., what routing protocols are used). Moreover, resources may be aggregated: instead of tracking and reporting sensor resources in detail, they may be summarized (e.g., coverage in an area instead of individual sensor location). Thus, the system must be able to associate and track resources at different granularities including resources that span multiple sensors.

A related issue arises due to the unique data-centric nature of sensor networks: they are embedded within the environment they are monitoring, and are mostly of interest only in terms of what information they can provide about the environment. Resources may be associated with regions in terms of network organization (e.g., resources or services in a cluster) or in terms of the environment being monitored (e.g., resources or services available in a room). The resource discovery protocol should provide the flexibility of tracking resources in terms of application-level organization or infrastructure-level organization.

The choice of what attributes to track and at what granularity to track them has to be made carefully by application developers and revised in response to observed use; we do not discuss this aspect of DRD further in this paper.

4.2 Resource Discovery Protocol

Once the tracked attributed set is determined, the role of the resource discovery protocol is twofold: (1) Track the values of the attributes at selected points within the network; and (2) Respond to client queries. An effective solution must take into account energy efficiency requirements. More specifically, it is necessary to minimize the communication required for implementing DRD. Energy efficiency also dictates being able to aggressively power down sensors when they are not being used; powered down sensors cannot receive or respond to queries.

In terms of how to track the selected attributes, a choice exists between pushing the attributes proactively to selected points in the network or pulling them in response to received queries. In distributed systems, generally pull is preferred to push when requests are infrequent relative to the data change rate, while push is preferred when requests are frequent. In sensor networks, two additional factors favor at least partial push of resource information: (1) because sensors must operate on a low duty cycle, pulling the data causes large delays if the data is locally maintained at sensors which are currently sleeping; and

(2) for complex attributes, the attributes must be summarized across multiple sensors; this requires pushing the data to points in the network where they can be combined.

Depending on the chosen approach, query resolution is implemented. In this component of the resource discovery protocol, the client query is forwarded to attribute repositories (locations in the network where attributes are collected) that are relevant to it. In a brute force unstructured approach, the queries may be flooded within the network. More efficient unstructured solutions may use gossiping or name-dropping [13]. Alternatively, if the resource space is structured, the location of the attributes that are relevant to a query are known then more efficient query forwarding can be used (e.g., name resolvers in DNS [22]). Further, caching may be used to optimize operation in either approach [29].

Ideally, the protocol should respond to resource queries in an energy efficient manner which requires minimizing the size of the exchanged messages. Interoperability favors a standardized protocol like ASCII-based XML that does not presume specialized protocols with potentially foreign elements generating the queries. However, efficiency dictates custom protocols that are compact. In the next two sections, we investigate alternatives to balance these requirements and show that it is possible to optimize the size of the exchange messages without sacrificing interoperability.

5 Architecture

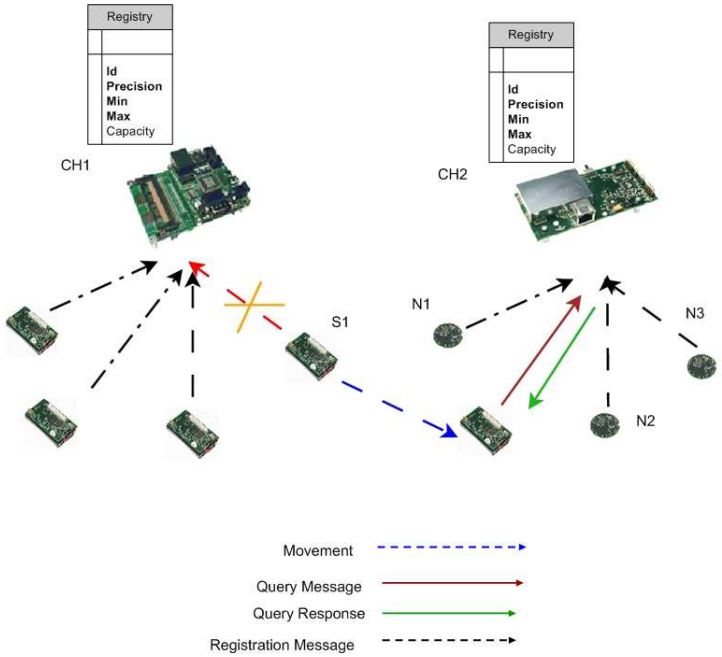

Fig. 2. Cluster based sensor network architecture.

To explore the feasibility of our vision of future sensor networks, we have developed a simulation-based prototype for DRD. We use an architecture where sensors self-organize to form clusters. A cluster is a collection of sensors that are associated/represented by a single *Cluster Head* (CH). For DRD, this organization serves the following purposes: (1) The CH represents a logical point for maintaining complex attributes; (2) The CH receives DRD queries and is able to respond to them, freeing the remaining sensors to be powered down in periods of inactivity; and (3) Finally, if hierarchical naming is desired, clusters can in turn themselves be clustered into bigger entities to provide more efficient query forwarding for structured attribute sets. We note that the use of clusters is common in sensor networks [15], mainly to allow data aggregation/reduction (similar to our complex attributes) and resource arbitration resulting in more scalable and energy efficient solutions.

Similar to GAF [39], in our clustering algorithm, cluster membership is determined geographically. More specifically, the sensor field is divided into zones such that all sensors within a zone are in range with each other. Cluster selection is then localized to a zone such that a sensor only considers CH advertisements occurring in its zone, therefore only one CH is selected per zone. We note that this approach requires either pre-configuration of the sensors or the presence of a location discovery mechanism (GPS cards or a distributed localization algorithm [4]). In sensor networks, localization is of fundamental importance as the physical context of the reporting sensors must be known in order to interpret the data. We therefore argue that our assumption that sensors know their physical co-ordinates is realistic. In any case, we emphasize that cluster formation is orthogonal to the proposed resource discovery protocols and other cluster formation approaches can be used.

The overall operation can be viewed as a continuous process consisting of the following phases:

1. Cluster head election: Cluster election is the process of identifying cluster heads, which serve as control points, data sinks and meta-data repositories for their cluster members. In this phase, each sensor sends an Resource Description Format (RDF) [36] advertisement indicating its remaining energy to its one hop neighbors. Upon receiving such messages from all of its one hop neighbors, each node then elects the node with the highest remaining energy as the CH for this round. The remaining nodes then attach themselves to the elected CH during the remaining phases. Note that we have chosen a simple CH criteria for our prototype, but that in practice it could be arbitrarily complex.
2. Meta-data exchange phase: During this phase, nodes send their meta-data to the CH as an RDF message, where it is inserted into an SQL database using a simple mapping from the RDF properties to table columns. For static attributes, this phase can be combined with CH election by piggybacking the meta-data on the CH election messages. However, for dynamic attributes meta-data collection might be required more frequently. For example, if a video sensor changes its angle, zoom, or resolution such meta-data needs to be registered each time the attributes change.
   Up till now, we have focused just on the discovery protocols (attribute tracking). However, meta-data is also needed to describe the data stream reported by a sensor. Interestingly, for some long running queries, the data reported by a sensor might change more dynamically. Consider a mote that has temperature, audio, and light sensor on board. The query might require the sensor to report its data with certain frequency if any of the temperature, light readings or audio signals have any interesting events. In such cases, the sensor needs to associate meta-data with its data stream so that the receiver can identify whether its a temperature, pressure, audio stream, or a combination of these streams.

3. Query generation: Consider a scenario where either a mobile client node (PDA) comes in contact with a foreign network or in the case of mobile sensors, a sensor might leave its current location and visit a new territory. For example, Figure 2 shows that a mobile temperature sensor S_1 moves out of range for CH-1 and moves in range with CH-2. In both case, the sensor itself might need to make use of various services available within its new environment (Cluster-2). For example, it might need to use a localization service to find out its new physical coordinates, and use a time synchronization service to adjust its clock. It might need to collaborate with other sensors in certain cases. Therefore S_1 needs to dynamically discover services and resources available within Cluster-2. S_1 then enters the query generation phase and sends a query message to CH-2. In our simulations, the query message is an SQL query sent as ASCII.
4. Query forwarding: In this phase, queries from clients are forwarded towards attribute repositories that can satisfy them. Depending on the application, there may be room for optimized query forwarding (e.g., for attributes that are strictly hierarchical, or when queries can benefit from the results of previous ones). For unstructured resources, multicast or epidemic algorithms will probably be needed to deliver queries.
5. Query processing: In this phase a CH upon receiving a query, processes it and sends back the appropriate reply.
6. Resource/service usage: Based on the reply from the previous stage, S_1 (or a client) starts accessing the requested resources and services. For example, a S_1 might start using the localization service provided by location anchors N_1 N_2. S_1 might negotiate the data transfer protocol and start reporting its observations to CH-2.

5.1 Discussion

The process of dynamic resource discovery involves several aspects, including resource description, resource registration, resource query, and message encoding. The above architecture describes the overall interaction of these aspects, but is not tied to the particulars of how resources are described, how queries are formulated, or how information is encoded. These can all be done in a number of ways, each with different trade-offs. Resource discovery in sensor networks can draw from a number of efforts in distributed systems and web services.

Resource description. A number of possibilities exist for resource description. We could of course simply use a completely ad hoc resource model. This has the advantage that we can customize it to be very compact, and tailored for DRD needs. The disadvantage, though, is that it would be incompatible with other resource descriptions. This means that it would not be able to leverage other associated software and standards. Since we could not see any significant advantage to creating our own resource descriptions, we used the RDF model developed by the W3C.

We anticipate that as our research matures, we will actually develop an ontology of resources. This will allow us to apply description logic to resources. As we develop an ontology, we will naturally move from RDF to OWL. This is aided by the fact that OWL actually uses RDF.

We are currently not using the standard RDF/XML syntax, but we anticipate that we will as our work matures. We have addressed size concerns by using binary XML, which we describe below.

Query formulation. When a client needs to query the CH for the available resources, it needs to formulate the query in some language. We are currently map the RDF description to a table, and thus formulate the query as an ASCII SQL string. This has the advantage of using a common, simple language for queries, but cannot handle more complex, structured resource queries.

Query formulation for resource descriptions and ontologies is currently an active research and development area. A number of RDF and OWL query languages exist, such as OWL-QL [10], RDQL [27], and RQL [19]. Further investigation is necessary to determine which approaches will work best for sensor networks.

Formats and encodings. A variety of formats can be used to encode the messages used in the various phases. Since communication requires large energy expenditures, a compact format has the advantage of conserving energy. For example, the following format might be used to represent the resources of a node.

```
41
1000
0
500
1000
```

This data being 5 integers, takes only 20 bytes, but to interpret these numbers correctly, the client must know that the first number is the ID, the second number is the precision, the third number is the minimum of the range, the fourth number is the maximum of the range, and the last number is the capacity. Such detail is error-prone, and the format may lack lack flexibility and interoperability. Changes to the format will be hard to detect, may cause strange failures or hard-to-find bugs.

Formats such as XML, along with standards such as SOAP [35] and WSDL [8], offer the advantage of having a large community of developers and researchers working on a common, well-known language. By using these, we can leverage these standards. Developers are likely already familiar with many of these, and thus do not have to learn yet another set of standards for networked communication.

The disadvantage of these formats, however, is that they are very verbose. An XML document typically contains many identical strings referring to tags, namespace prefixes, and namespace URIs. Every XML element also includes a redundant end tag. The following XML document contains the same data as the custom message above, but requires 191 bytes.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<index version=''1.0''>
  <id>41</id>
  <precision>100</precision>
  <minrange>0</minrange>
  <maxrange>500</maxrange>
  <capacity>1000</capacity>
</index>
```

Using XML has a number of attractive benefits, but the verbosity incurs large energy costs during communication. One alternative that reduces the energy costs, but retains many of the advantages, is to use alternative encodings of XML.

XML is specified as a textual format for structured data. Beyond syntax, however, the specification also implicitly suggests an abstract data model for tree structured data consisting of parent nodes and child nodes, with their associated attribute, content, and other information. This model does not inherently preclude an efficient encoding, and thus alternate serializations of it can have much more concise than textual XML. These alternative serializations, are commonly known as "binary" XML, since they often utilize the full 8-bit range of the constituent bytes.

The use of binary XML is a compromise between textual XML and custom formats. Binary XML is still in its early stages, and there is no widespread standard for it. However, since binary XML is logically the same as textual XML, a single API can work for both. Also, all standards based on XML will also work with binary XML, since most XML standards reference the XML abstract model known as XML Infoset.

We have tested two different versions of a binary XML (details pending in a separate publication). Our first version eliminates the redundant end tag, and replaces ASCII numbers with their two's complement representations. Though this is considerably more compact than textual XML, it does not begin to approach the compactness of a custom format.

The second version achieves compactness by separating much of the static meta-data about the message from the data of the message itself. This is achieved by using a separate XML Schema of the message, and combining the Schema with a compact representation of the data to reconstitute the complete XML document. This is similar to ideas in PBIO [9] and DFDL [6].

Interpretation of these compact messages requires an associated schema. This association can occur through a number of mechanisms. One technique is to simply include a schema URI in the beginning of the message. This overhead may be too high for short messages, however, since the URI is typically a relatively long string. Another technique is to assume that some previous information has been exchanged, and then associated with the particular communication channel corresponding to the compact message. For our prototype, we have simply hard-coded this association. We note that this use of the schema to interpret a compact representation of the message can also be applied to textual XML.

6 Experimental Study

In this section we first describe the design of our simulation framework and then demonstrate its utility using a simple prototype simulation study. We hope to extend this framework as we develop our VSN subsystem to complement implementation on real sensors.

6.1 Simulation Testbed

We now present our evaluation testbed, which is completely based on open source software components; either already available in public domain or built in house. The integration of these typical software components ranging from an embedded database, network simulator, and XML parsers provides an accurate development and evaluation environment that can reduce the development barriers for a broad spectrum of applications. It can also assist researchers in evaluating their protocols. We now describe the individual components of the framework.

– *SQLlite database.* SQLite is a self-contained, embeddable, zero-configuration SQL database engine [28]. We decided to choose SQLite to track the resource attributes (and potentially sensor data) because it has following interesting features. It implements most of

SQL92 and requires zero-configuration – no setup or administration is needed. This property is useful because the large scale and autonomous nature we target, prohibits manual configuration and administration. It stores the complete database in a single disk file. This is important given the resource constraints – having multiple database files on disk and memory may be possible for some embedded sensor filesystems. Database files can be freely shared between machines with different byte orders. Most important to our purposes, it has a small code footprint and the sources are in the public domain.
- *Libxml2*. It is the XML C parser [33] and toolkit developed for the Gnome project and the sources are available in public domain. We are also exploring various other XML parsers such as TinyXml [31], XPP [38], Xerces [37] and would like to evaluate their performance and power characteristics.
- *Binary xml parser*. We used an experimental binary XML implementation developed in-house, which is discussed in Section 5.1.
- ns-2: It is an open source discrete event simulator used for simulating networking details [23]. The front end of simulator is written in TCL, whereas its back-end is written in C++ mainly for performance reasons. In our simulations we used a CSMA based MAC layer and our power analysis is based on the energy model pre-built in ns-2.

This framework does not imply that there exists heterogeneity in terms of software elements. Specifically, we do not expect individual software components (SQLite, libxml etc.) to be installed on all the sensors. However, we expect the system components to follow appropriate standards and expose standardized interfaces. For example, a mote might run an instance of TinyDB [30] database, whereas a PASTA node might run an instance of SQLite. However, since both these database technologies follow standards and expose SQL interface, they fit in within our system design notion. To summarize, we take a technology agnostic view and our system design is based on open standards and standardized interfaces. Above mentioned software components just provice a concrete basis to build our simulation framework, and not as an end in itself.

At some level in the system, commonality must exist. From this common level, clients would then bootstrap and configure themselves to work with the relevant sensors. For the lowest levels, we believe that RDF is the appropriate choice. Above that, however, we believe that the issue is an open question which we will address in future work.

We also believe that connectivity to the grid and web services will be important. Such interoperability can greatly increase the effectiveness and utility of sensor networks by allowing them to be plugged into wide area cyberinfrastructures.

6.2 Simulation Results

To demonstrate the utility of the proposed framework, we conducted prototype study to evaluate the energy efficiency of message encoding format alternatives. Table 2 describes the parameters used in our simulations.

For modeling energy model realistically, we used a 802.15.4 style low power radio in our simulations. The simulation area was set to $350 \times 350\ m^2$ and each zone was $70 \times 70\ m^2$ (total 25 zones). Radio transmission range was set to $100\ m$ to ensure that all sensors within a zone are in range with each other.

Table 3 shows the mean energy consumed per sensor per message for various message encoding formats. As expected, we see that using ASCII based XML format, both the transmission and reception energy consumed is almost 9 times higher than that of custom

Table 2. Simulation parameters.

Simulation area	$350 \times 350\ m^2$
Transmission range	$100\ m$
Initial-Energy	$10000\ J$
MAC Protocol	$CSMA - CA$
Bandwidth	$250\ kbps$
Transmit Power	$0.0522\ W$
Receive Power	$0.0591\ W$
Idle Power	$0.00006\ W$
Number of Nodes	50

encoding approach. On the other hand the use of a compact binary XML encoding format (with predetermined, separated XML schema) consumes same amount of energy as that of a custom encoding format. As explained in Section 5, completely customized protocols are most energy efficient but offer very little flexibility and interoperability. Whereas the textual XML represents the other end of the spectrum. Using XML has a number of attractive benefits, but the verbosity incurs large energy costs during communication (almost 9 times higher than the custom based formats). However, the use of binary XML is a compromise between textual XML and custom formats.

The results helped us to validate our simulation framework.

Table 3. Energy consumption study.

Protocol	Data size (bits)	Transmit Energy (J)	Receive Energy (J)
custom	160	0.000033	0.000038
ASCII xml	1448	0.000302	0.000342
binary xml	160	0.000033	0.000038

7 Related Work

In an open distributed system, the problem of naming and locating resources is challenging and has been well studied in literature [2, 13]. Several protocols for service discovery have been proposed [17, 32, 34]. Moreover, in mobile environments, the effect of mobility has been also considered (e.g., [5, 25]).

The primary design goal of Jini [17] is network plug and play and it supports service discovery via service registration and lookup. While Jini works well for its intended applications, we believe that it is not ideal for realizing a DRD-based vision of sensor networks. First, Jini is Java-based, and this permeates its design. In theory, one could bind its wire formats and protocols to other languages, but the result would likely be awkward. For example, Jini's service description model is based on Java's rules for class derivation. A service

query matches a service if it is a supertype or same type as the service. This categorically precludes the use of a number of promising research areas for wide-scale interoperability and mediation, such as ontologies, RDF query languages [27, 19], description logics [3], and semantic mediation [12].

Its communication model is based on Java RMI and it relies heavily on passing Java objects across network. This can impose high energy costs for small objects, which will be typical for DRD. Jini was not designed for WSNs. Though it could clearly be adapted and modified to work well, such changes would be extensive, and the resulting architecture would lose most of the advantages of using Jini in the first place. It would not be compatible with Jini, and would not be able to leverage other technologies built with Jini. For these reasons we believe that Jini's protocol architecture is heavy weight for resource constrained sensors.

Universal Plug and Play (UPnP) architecture [32] also aims for zero-configuration. However, it has been designed for a different context. In UPnP, components are divided into control points and devices. Each device has a device service. When a device comes on-line, it broadcasts itself to control points on the network. Once a control point has discovered a device, it then takes control of the device and sents control messages to the device's service. UPnP is for a network to discover devices, rather than for a client to discover resources, and does not include any notion of clustering. It is also designed for resource rich devices and its complex architecture is not easy to deploy on resource constrained sensors.

Additionally, many existing resource discovery schemes including Jini [17], UPnP [32], and SDS [5] assume an underlying IP based networking infrastructure. They also rely on support for multicast and assume presence of a reliable transport layer protocol such as TCP. However, the communication models as well as protocol stack architecture for sensor networks is still evolving and it is not clear whether the final model will support an IP based communication infrastructure or employ a complex machinery such as TCP. Therefore we believe that the existing technologies are not directly applicable to sensor networks.

Approaches developed in the context of ubiquitous computing [2] and mobile computing [5] share some of our design goals in terms of ubiquitous inter-operation and energy efficiency respectively. However, they do not take advantage of other properties of sensor devices and applications (such as data-centric operation, and different levels and modes of granularity) to realize their systems.

Recently, Stann and Heidemann proposed resource discovery optimizations for sensor networks [29]. In this work, a homogeneous sensor network is assumed (at least, in terms of administration and software). In addition, resource discovery is integrated with the directed diffusion model which is used for data collection [16]; queries are forwarded towards data sources simulataneously discovering and reserving required resources in a position to report required data. The target of this work is to optimize this flooding process by taking advantage of historical queries for similar resources. The scope of our work is wider (not just query forwarding) and is not tied to the directed diffusion model.

8 Concluding Remarks

Resource discovery is an important first step towards enabling interoperability of sensor networks, and eventually seamless integration among them (what we call *Virtual Sensor Networks*). In this paper, we define the resource discovery problem in sensor networks and outline the challenges involved in it. Sensor networks are unique in several respects that influence resource discovery and necessitate specialized solutions. More specifically, their data-

centric embedded nature, relatively poor resources, the emphasis on energy efficiency and the lack of existing standards combine to render traditional resource discovery approaches ineffective.

The paper first explores the space of the resource discovery problem in sensor networks. Resource discovery was organized into two complementary components: the decision on what attributes of the system to track; and the design of energy efficient and available resource discovery protocol. We outlined the challenges involved in each of the subsections. In addition, we demonstrated a solution to one of the challenges (the balance between interoperability and efficiency in the resource discovery protocol) and showed that its possible to improve efficiency while maintaining interoperability.

In Section 6.2 we described our simulation framework and presented preliminary simulation results. The simulation framework can be used to characterize the performance of various protocols on large-scale sensor networks. However, the real value of our research can be realized in the context of real sensor networks. To that end, we are currently working on implementing the proposed protocols on a real sensor network testbed. Our testbed consists of Mica-2, PASTA, and stargate nodes.

We are also exploring the concept of Virtual Sensor Networks in the context of second generation sensor networks. More specifically, we are studying various challenges associated with them. To that end, we would like to propose a component-based, modular, efficient service-oriented architecture.

References

1. H. Abrach, J. Carlson, H. Dai, J. Rose, A. Sheth, B. Shucker, and R. Han. Mantis: System support for multimodal networks of in-situ sensors. In *Technical Report CU-CS-950-03, Department of Computer Science, University of Colorado*, April 2003.
2. W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley. The design and implementation of an intentional naming system. In *Symposium on Operating Systems Principles*, pages 186–201, 1999.
3. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider.
4. N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low cost outdoor localization for very small devices. *IEEE Personal Communications Magazine*, Oct. 2000.
5. S. E. Czerwinski, B. Y. Zhao, T. D. Hodes, A. D. Joseph, and R. H. Katz. An architecture for a secure service discovery service. In *Mobile Computing and Networking*, pages 24–35, 1999.
6. Data Format Description Language project web page. `http://forge.gridforum.org/projects/dfdl-wg/`, 2004.
7. Habitat monitoring on great duck island. `http://www.greatduckisland.net/`.
8. E. Christensen et. al. Web Services Description Language (WSDL) 1.1. `http://www.w3.org/TR/wsdl`, 2001.
9. G. Eisenhauer and L. K. Daley. Fast heterogenous binary data interchange. In *Proceedings of the Heterogeneous Computing Workshop (HCW2000)*, 2000.
10. R. Fikes, P. Hayes, and I. Horrocks. OWL-QL: A language for deductive query answering on the semantic web. `ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/KSL-03-14.pdf.gz`, 2003. KSL Technical Report 03-14.
11. I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150:1–??, 2001.
12. A. Gupta et al. Registering Scientific Information Sources for Semantic Mediation. In *21st International Conference on Conceptual Modeling*, 2002.
13. M. Harcol-Balter, P. Leighton, and D. Lewin. Resource discovery in distributed networks. In *Proc. of ACM PODS 1999*, pages 229–237, 1999.

14. T. He, J. A. Stankovic, C. Lu, and T. Abdelzaher. Speed: A stateless protocol for real-time communication in sensor networks. In *ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems*, page 46, Washington, DC, USA, 2003. IEEE Computer Society.
15. W. Heinzelman. *Application-Specific Protocol Architectures for Wireless Networks*. PhD thesis, Massachusetts Institute of Technology, 2000.
16. C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proc. 6th ACM International Conference on Mobile Computing and Networking (Mobicom'00)*, Aug. 2000.
17. Sun microsystems. jini technology architectural overview (white paper). `http://www.sun.com/software/jini/whitepapers/architecture.html`.
18. P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. In *In Prof. of ASPLOS 2002*.
19. G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, and M. Scholl. Rql: a declarative query language for rdf. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 592–603, New York, NY, USA, 2002. ACM Press.
20. C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He. Rap: A real-time communication architecture for large-scale wireless sensor networks. In *RTAS '02: Proceedings of the Eighth IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'02)*, page 55, Washington, DC, USA, 2002. IEEE Computer Society.
21. Crossbow – smart sensors in silicon (crossbow website), 2003. Commercial product based on Berkeley MICAs `http://www.xbow.com`).
22. P. Mockapetris and K. J. Dunlap. Development of the domain name system. In *SIGCOMM '88: Symposium proceedings on Communications architectures and protocols*, pages 123–133, New York, NY, USA, 1988. ACM Press.
23. Network Simulator. http://isi.edu/nsnam/ns.
24. Pasta microsensor user's guide. (`http://pasta.east.isi.edu/documentation/users_guide/`).
25. C. Perkins and H. Harjono. Resource discovery protocol for mobile computing. *Mobile Networks Journal*, 1(4):447–455, 1996.
26. Rockwell science center sensor network project, 2003. (`http://www.rsc.rockwell.com/wireless_systems/sensorware`).
27. A. Seaborne. Rdql - a query language for rdf. `http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/`, 2004.
28. Sqlite. `http://sqlite.org/`.
29. F. Stann and J. Heidemann. BARD: Bayesian-assisted resource discovery in sensor networks. In *Proceedings of the IEEE Infocom*, 2005.
30. Tinydb: In-network query processing in tinyos. `http://telegraph.cs.berkeley.edu/tinydb/doc/index.html`.
31. Tinyxml. `http://sourceforge.net/projects/tinyxml/`.
32. Universal plug and play device architecture. `http://www.upnp.org/download/UPnPDA10_20000613.htm`.
33. D. Veillard. Libxml2 project web page. `http://xmlsoft.org/`, 2004.
34. J. Veizades, E. Guttman, C. Perkins, and S. Kaplan. Service location protocol, 1997.
35. W3C. Simple Object Access Protocol (SOAP) 1.1. `http://www.w3.org/TR/2000/NOTE-SOAP-20000508/`, 2000.
36. W3C. Resource description framework (rdf). `http://www.w3.org/RDF/`, 2004.
37. Xerces: Xml parsers. `http://xml.apache.org/#xerces`.
38. Xml pull parser. `http://www.extreme.indiana.edu/xgws/xsoap/xpp/`.
39. Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the 7th annual international conference on Mobile computing and networking*. ACM Press, 2001.

40. W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks, 2002.