PRESTO: A Predictive Storage Architecture for Sensor Networks[†]

Peter Desnoyers, Deepak Ganesan, Huan Li, Ming Li, Prashant Shenoy
University of Massachusetts Amherst

Abstract—**We describe PRESTO, a predictive storage architecture for emerging large-scale, hierarchical sensor networks. In contrast to existing techniques, PRESTO is a proxy-centric architecture, where tethered proxies balance the need for interactive querying from users with the energy optimization needs of the remote sensors. The main novelty in this work lies in extensive use of predictive techniques that are a natural fit to the correlated behavior of the physical world. PRESTO exploits technology trends in storage to build an architecture that emphasizes archival at remote sensors and intelligent caching at proxies. The system also addresses user needs for querying such sensor networks by exposing a unified, easy to use data abstraction across numerous proxies and remote sensors.**

I. Introduction

Many different kinds of networked data-centric sensor systems have emerged in recent years. Sensors generate data that must be processed, filtered, interpreted, cached, and archived in order to provide a useful infrastructure for users. Sensors are often untethered, and their energy resources need to be optimized to ensure long lifetime [2], [13]. Thus, energy-efficient data management is a key problem in sensor applications.

There are two commonly used models for processing data in sensor networks. The first model involves viewing the sensor network as a database [1], [2], [3], where queries are pushed all the way to the remote sensors. Such direct querying of the remote sensor nodes is generally more efficient energy-wise, since query-specific data processing can be performed at the data source to reduce communication requirements. However, such querying renders the system unusable for interactive use due to the high latency, low availability, and low reliability [4] inherent in duty-cycled, energy-limited wireless sensor networks. The second model has been one of data streams, where potentially useful sensor data is pushed from the sensors, and stored at a high-end server running a database. The database engine can perform statistical modeling and cleaning on the data [5], and provide lower latency, better availability, and better interactivity to user queries. However, this model is less energy efficient since it does not exploit the fact that only a subset of sensor data may be actually queried.

While both these models are important for current and future sensor networks, they have certain drawbacks. In this paper, we present PRESTO, a predictive store for sensor networks that attempts to provide the interactivity of the data streaming approach with the energy efficiency of the direct sensor querying. PRESTO differs from past work on data-centric sensor networks in several key respects (see Table I).

- **Hierarchical Systems:** Rather than designing our system for a single flat sensor network architecture, PRESTO reflects our philosophy that scalable sensor networks of the future will have multiple tiers, with a several tens of untethered sensors per tethered sensor proxy and several tens of sensor proxies per application. Being tethered, sensor proxies can be expected to be less resource constrained than the remote sensors, an aspect that PRESTO exploits in two different ways. Proxies cache current and past data from remote sensors and use predictive techniques on cached data to answer queries, thereby providing response times that are close to the data streaming approach. Proxies also use their superior processing capabilities to model, predict, and match query parameters to data dissemination at remote sensors, thereby providing the energy efficiency of the direct querying method.
- **Archival Queries:** Unlike many systems that only support queries on the current sensor data [5], PRESTO supports archival queries on data that may be deemed to be interesting *post-facto*. The ability to query historical data is important in many sensor applications such as surveillance, where the ability to retroactively "go back" is necessary to determine, for instance, how an intruder broke into a building. Similarly, archival sensor data is often useful to conduct postmortems of unexpected and unusual events to better understand them for the future. PRESTO enables such PAST queries by employing a distributed archival store at remote sensors that records past sensor data; thereby resulting in a significantly different architecture from stream-based systems.
- **Single Logical View of Data:** PRESTO goes beyond techniques that have focused on a single wireless sensor network deployment, and aims to provide a single logical view of data distributed across many sensor proxies and numerous remote sensors. Such a view abstracts the user

† This work was supported by a grant from the Engineering Research Centers program of the National Science Foundation under cooperative agreement EEC-0313747.

TABLE I

COMPARISON OF PRESTO TO RELATED EFFORTS.

	NOW Queries	PAST Queries	Prediction	Data Abstraction	Energy-Aware	Proxy-Sensor Interaction	Hierarchical design
Diffusion[2]	Direct sensor querying	No archival	No	Single remote sensor	Yes	No	No
Cougar[1]	Direct sensor querying	No archival	No	Single remote sensor	Yes	No	No
TinyDB[6]/BBQ[5]	Proxy querying	Archival at proxy	Yes	Single proxy	Yes	No	No
Aurora/Medusa[7]	Proxy querying	Archival at server	No	Distributed stream	No	No	P2P
PRESTO	Proxy querying + sensor querying on cache miss	Caching at proxy + archival at sensor	Yes	Single logical view of distributed store	Yes	Yes	Yes

from the variabilities at many levels—lossy and unreliable remote sensor network; spatial and temporal consistency issues in the sensor data; as well as bandwidth and connectivity issues in the case of wireless proxies.

The novelty of PRESTO lies in its predictive storage capabilities and active interactions between proxies and sensors. Unlike traditional storage systems that are passive, the PRESTO proxy employs an active cache that predicts data values that are yet to be fetched from remote sensors (and thus, yet to be written to the local cache). While predictive techniques are also used in BBQ [5], we differ in that PRESTO uses active interactions to handle the occasional rare events that are inherently unpredictable. The PRESTO proxy provides feedback to remote sensors that limits the communication overhead of the sensor to data that is deemed "unpredictable" at the proxy. Such a predictive push-based approach ensures that rare, unexpected events are never missed, which is important in many event-driven applications such as intruder detection. When cache misses occur at the proxy, PRESTO reverts to direct querying of data archives at remote sensors.

Several technology trends make such a predictive storage architecture both feasible and appealing. First, radio communication is generally considered to be quickly reaching fundamental energy barriers [8]. Hence, the commonly held view is that communication should be reduced and compensated by increased use of either computation (up to four orders of magnitude less expensive [8]) or storage (two orders of magnitude less expensive[1]). Second, capacities of flash memories continue to rise as per Moore's Law, and their costs continue to plummet. Thus, it will soon be feasible to build cost-effective sensor nodes with up to a gigabyte of flash memory. PRESTO exploits the presence of a large local store to reduce communication by archiving data locally at remote sensors whenever possible. Finally, processing speeds continue to increase, with new energy-efficient technologies delivering more CPU cycles per watt. This enables us to put more capable processors on remote sensors as well as intermediate

[1]Comparing 1Gb Samsung NAND Flash and MICAz ZigBee radio

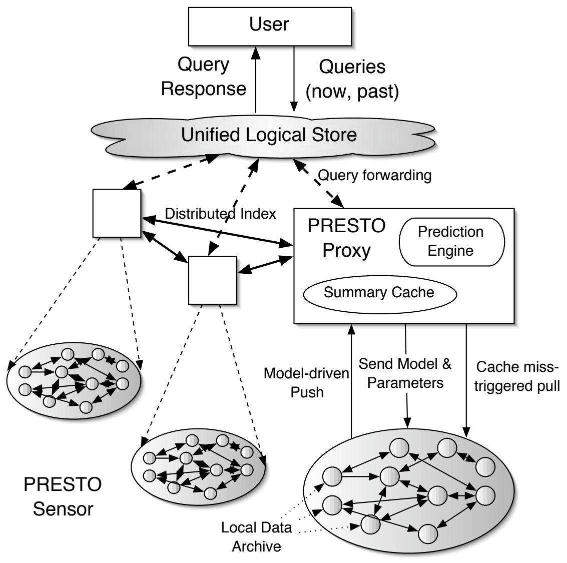

Fig. 1. PRESTO architecture

proxies and leverage the additional processing capacity for extrapolation, batching, and compression, all of which can reduce communication.

The rest of this paper is structured as follows. Section II describes our system architecture. Sections III, IV, and V describe the PRESTO proxy, sensor and the data abstraction, respectively. We conclude in Section VI.

II. SYSTEM ARCHITECTURE

Our view of the emerging sensor network architecture comprises three tiers as shown in Figure 1—a bottom tier of untethered remote sensor nodes, a middle tier of tethered sensor proxies, and an upper tier of user terminals.

The lowest tier is assumed to form a dense deployment of low-power sensors. A canonical sensor node at this tier is equipped with low-power sensors, a micro-controller, and a radio as well as a significant amount of flash memory (1GB). This tier may be heterogeneous, and might comprise different kinds of devices, sensors and platforms. In the future, some limited form of energy harvesting might assist these sensors in achieving even greater lifetimes than is currently achievable. The common constraint for the lowest tier is energy, and the need for a long lifetime inspite of it. The use of radio, processor, RAM, and the flash memory all consume energy, which needs to be limited.

The middle tier consists of power-rich sensor proxies that have significant computation, memory and storage resources and can use these resources continuously. Many different instances of this middle tier can be seen in different application settings. In urban environments, this tier would comprise a tethered base-station class node (e.g., Intel Stargate) with multiple radios—an 802.11 radio that connects it to a wireless mesh network and a low-power radio (e.g. Zigbee) that connects it to the sensor nodes. Since Internet connectivity is widely available in many urban settings, these proxies may plug in to existing mesh networks or the wired infrastructure. In remote sensing applications [9], this tier could comprise a similar Stargate node with a solar power cell. Each proxy is assumed to manage several tens of lower-tier sensors in its vicinity. A typical sensor network deployment is will contain multiple geographically distributed proxies. For instance, if a building is being monitored, one sensor proxy might be placed per floor or hallway. At the highest tier of our infrastructure are users, who can query the sensor network through a query interface, perhaps using declarative queries as proposed in TinyDB [6].

System Operation: Although the PRESTO architecture does not preclude continual queries, in this paper, we focus on the mechanisms needed to support one-time queries on current and past sensor data. Each proxy is assumed to cache data summaries or a subset of the data from the lower tiers sensors. When a new query arrives, the proxy examines its cache to see if the data necessary to answer the query is available. In the event of a hit, the query can be processed locally. Cache misses are handled in one of two ways. The proxy first examines other cached data to see if the requested data can be extrapolated from it. Cached data from other nearby sensors or temporally adjacent data from the sensor can be used for such extrapolation, and the extrapolated data can be used to process the query locally. If the spatio-temporal extrapolation does not yields sufficiently accurate data to meet the query error tolerances, then the cache miss is handled by fetching data from other sensor caches or the archive at remote sensors. This is enabled by a complete local archive of past data at each remote sensor. On storage-constrained sensors, older archived data is aged gracefully to ensure that lower resolution representations are available [10].

To ensure that all "interesting" data is cached at a proxy with high probability, PRESTO employs a model-driven push approach. The prediction engine at the proxy builds models of correlations in the data and periodically transmits parameters of this model to the remote sensors. The remote sensors check their sensed data against this model and push data solely when the model fails, thereby saving energy-intensive communication at the expense of some cheaper computation. Such a model-driven push ensures that the proxy is notified of all significant drifts in sensor values as well as unusual changes caused by unexpected events. Observe that a pure pull-based approach can handle the former case but will likely fail to capture the latter scenario. In addition to an energy-efficient model-driven push, the PRESTO prediction engine also utilizes query characteristics such as query type, arrival rate, latency, and precision requirements to extract additional energy savings. For instance, sensors can be adaptively duty cycled and can employ batching to reduce their energy needs.

In the following sections, we describe the components of PRESTO in greater detail.

III. PRESTO Proxy

The PRESTO proxy comprises two components: a cache of summary information about the data observed at the remote sensors and a prediction engine that is responsible for data extrapolation, model-driven push, and query-sensor matching.

Sensor Data Cache: A central component of the sensor proxy is a summary cache of the data from remote sensors. This cache differs significantly from both memory caches as well as web caches in that the cached data is either a lossy view or a higher-level semantic event-based view of the sensor data. For instance, rather than sending the full data, sensors may transmit summaries of their observations to the proxy cache. Similarly, rather than sending raw data, a sensor may send processed events to the proxy. To illustrate, a camera sensor in a surveillance application may send notification that a new object has been detected and its type, rather than sending a raw image of the object. Such lossy or semantic representations of the data not only incur a smaller communication cost, they may be more appropriate from an application perspective. Further, the summary data cache at the proxy can be progressively refined as more accurate data is obtained from the remote sensors or as queries on past data results in missing portions of the cache being filled up.

Prediction Engine: The prediction engine at the proxy uses its prediction capabilities for three purposes: model-driven push, data extrapolation and query-sensor matching.

Model-Driven Push: PRESTO uses predictive modeling to enable model-driven push from the remote sensors. To do so, the proxy constructs a model that captures expected variations in the data and transmits parameters of this model to each remote sensor. For instance, a model of temperature variations will capture time-of-day effects (such as [5]) as well as the impact of seasons. Each remote sensor checks their sensed data against this model and transmits solely when the model fails, thereby saving energy-intensive communication at the expense of some cheaper computation. For instance, only deviations from the normal temperature for each hour of the day are reported. We seek a few important characteristics from these models. First, we require that models be *asymmetric*—they can be hard to build at the proxy, but they must require little resources to verify at the sensor. Thus, sensors must expend as little processing as possible to check if the sensed data conforms to the model. Second, the models should effectively capture the statistics of the underlying physical process corresponding to the sensor data. For instance, simple regression techniques and time-series analysis techniques may be used to model many temporal phenomena. Similarly, like in the acquisitional query processor work [5] a combination of multivariate models for the spatial axis and markov model for the temporal axis can also be used for model weather data.

Extrapolation: The PRESTO prediction engine can also extrapolate missing data that are needed by a query. As explained earlier, extrapolated data can mask cache misses and answer queries so long as the query precision is met. Observe that the above predictive data models can serve the dual purpose of enabling data extrapolation at the proxy, while dictating which data needs to be pushed by the remote sensors. For instance, in the absence of failures and even when sensors do not report any observations, it is safe to assume that the temperature at a certain hour or the day conforms to the historical trends captured by the model. These values can be substituted for the actual observations and used to answer queries. Thus, data extrapolation enables the proxy to provide quick and accurate responses to queries even if the data corresponding to the query is missing from the cache. Our work builds on existing techniques such as multivariate data modeling proposed in TinyDB and BBQ [5].

Query-Sensor Matching: Finally, the PRESTO prediction engine is responsible for query-sensor matching to match the needs of queries to the operations of remote sensors. To maximize savings, sensors can be adaptively duty cycled and asked to batch and compress a set of data values prior to transmission. The proxy takes into account the characteristics of queries for such matching-based optimizations. The query type, frequency, latency and precision requirements are translated into the appropriate parameters for the remote sensors, such that they can minimize energy while achieving query requirements. For instance, if it is known that the worst case notification latency for typical queries is 10 minutes, the proxy can instruct remote sensors to set its radio duty-cycling parameters accordingly in order to conserve energy. The duty cycling parameters can be adaptively varied as new queries with different needs arrive into the system. Similarly, if the queries only require 75% precision in their response, lossy compression and aggregation techniques can be used to reduce the amount of transmitted data. The type of query can be exploited as well. For instance, scientists studying building health monitoring are typically interested in the mode of vibration of a building. The operation can be transmitted as a parameter to the sensor node, which uses the specified mode function on its local data before transmitting the final result.

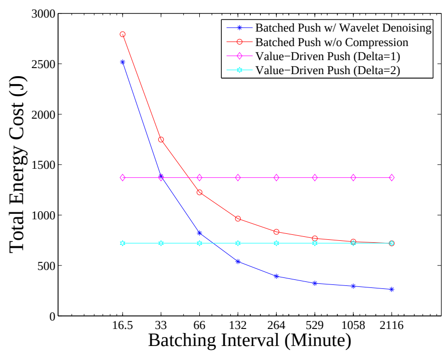

Fig. 2. Exploiting batching to conserve energy

Figure 2 shows one instance of such query-sensor matching in the case of temperature data [11], where the impact of batching on overall energy savings is demonstrated. Greater batching translates into two energy gains: (a) fewer packets imply a lower per-packet overhead including ACKs, packet headers and MAC-layer preambles, and (b) more batching results in better compression and data cleaning at the source of data, in this case, using wavelet denoising [12].

IV. PRESTO Sensor

PRESTO is a proxy-centric architecture where much of the intelligence resides at the proxy, and the remote sensor is kept simple to enable efficient operation under resource constraints. Our contribution lies in the design of sensors that are simple, yet highly tunable and can be completely controlled by the proxy. The PRESTO sensor has two components. The first is an archival file-system that we are developing, called *WriteHere*, that provides energy-efficient archival of data at each sensor as well as a simple time-based index structure to efficiently service read requests. Data archival at the remote sensors is needed to deal with queries on past data that may not be cached at the proxy. If storage is constrained on each sensor, graceful aging of data can be enabled using wavelet-based multi-resolution techniques [10]. The second component is a simple adaptive system that can use the information provided by the proxy to tune data transmission, data processing, aggregation, as well as duty-cycling parameters. For instance, the sensor node may aggregate data using wavelet compression if queries require time-series summaries of data, but can switch to fourier transforms if the frequency content is queried.

V. PRESTO Data Abstraction

A key goal of PRESTO is to provide a unified data abstraction of a single logical store across tens to hundreds of proxies and thousands of remote sensors that comprise a sensor application. Part of this abstraction is enabled by the sensor proxy that abstracts the user from vagaries of the remote sensor network. The second enabling system component is a distributed index structure that constructs a unified view of caches across geographically distributed sensor proxies.

The PRESTO data abstraction has three goals. The first is to provide a single temporally ordered view of the data across different proxies. To enable such a view, the system needs to deal with temporal consistency issues, since drift and skew of clocks at the remote sensors can result in erroneous timestamps. In our current work, we are exploring the use of order-preserving index structures such as Skip Graphs [14] for this purpose. The second goal is dealing with spatial consistency issues that arise due to overlapping coverage areas between proxies. Multiple proxies might be responsible for a group of sensor nodes for redundancy, reliability, and fault-tolerance reasons, and hence, cache consistency issues need to be addressed. Finally, the index structure will span a mix of wired sensor proxies with high bandwidth links and wireless 802.11-based proxies with lower bandwidth and availability. Cached data at the wireless proxies may need to be aggressively cached at the wired proxies to enable low-latency query responses in spite of the vagaries of 802.11 links.

VI. Discussion and Conclusions

We described PRESTO, a predictive storage architecture for emerging large-scale, hierarchical sensor networks. In contrast to existing techniques, PRESTO is a proxy-centric architecture, where tethered proxies balance the need for interactive querying from users with the energy optimization needs of the remote sensors. The main novelty in this work lies in extensive use of predictive techniques that are a natural fit to the correlated behavior of the physical world. PRESTO exploits technology trends in storage to build an architecture that emphasizes archival at remote sensors and intelligent caching at proxies. The system also addresses user needs for querying such sensor networks by exposing a unified, easy to use data abstraction across numerous proxies and remote sensors.

PRESTO can be used in different ways in different application contexts. Environmental weather patterns and commuter traffic patterns are examples of data that are highly predictable in the common case. PRESTO can enable the system to conserve energy by learning the predictable aspects of the data, and efficiently extracting only the unpredictable information from remote sensors. The unified data abstraction and predictive responses that PRESTO provides can be used in vehicle traffic querying as commuters can query the system to obtain quick responses. Surveillance applications can use the archival capability of PRESTO to query for event logs corresponding to past events.

References

[1] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems," in *Proceedings of ICMDM*, Hong Kong, January 2001.

[2] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proceedings of the ACM/IEEE Mobicom*, Boston, MA, USA, Aug. 2000.

[3] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, L. Yin S. Shenker, and F. Yu, "Data-centric storage in sensornets," in *ACM Hotnets*, 2001.

[4] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker, "Complex behavior at scale: An experimental study of low-power wireless sensor networks," Tech. Rep. UCLA/CSD-TR 02-0013, UCLA, Department of Computer Science, 2002.

[5] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *VLDB*, 2004.

[6] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," in *OSDI*, Boston, MA, 2002.

[7] H. Balakrishnan, M. Balazinska, D. Carney, U. Cetintemel, M. Cherniack, C. Convey, E. Galvez, J. Salz, M. Stonebraker, N. Tatbul, R. Tibbets, and S. Zdonik, "Retrospective on Aurora," *VLDB Journal*, vol. 13, no. 4, 2004.

[8] G. Pottie and W. Kaiser, "Embedding the internet: wireless integrated network sensors," *CACM*, vol. 43, no. 5, pp. 51–58, May 2000.

[9] L. Girod, T. Stathopoulos, N. Ramanathan, J. Elson, D. Estrin, E. Osterweil, and T. Schoellhammer, "A system for simulation, emulation, and deployment of heterogeneous sensor networks," in *Proceedings of ACM Sensys*, Baltimore, MD, 2004.

[10] D. Ganesan, B. Greenstein, D. Perelyubskiy, D. Estrin, and J. Heidemann, "Multi-resolution storage in sensor networks," in *Proceedings of ACM Sensys*, 2003.

[11] P. Bodik, W. Hong, C. Guestrin, S. Madden, M. Paskin, and R. Thibaux, ," Intel Lab Data. `http://db.lcs.mit.edu/labdata/labdata.html`.

[12] M. Vetterli and J. Kovacevic, *Wavelets and Subband coding*, Prentice Hall, New Jersey, 1995.

[13] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," in *Proceedings of ASPLOS-IX*, Cambridge, MA, USA, November 2000.

[14] G. Shah J. Aspnes, "Skip graphs," in *SODA*, Jan 2003.