Energy Enhancement of Multi-application Monitoring Systems for Smart Buildings

Ozgun Pinarer[1,2(✉)], Yann Gripay[1], Sylvie Servigne[1], and Atay Ozgovde[2]

[1] Universite de Lyon, CNRS INSA-Lyon, LIRIS, UMR CNRS 5205, 69621 Lyon, France
{ozgun.pinarer,yann.gripay,sylvie.servigne}@insa-lyon.fr

[2] Department of Computer Engineering, Galatasaray University, Ciragan Cad. No: 36, 34349 Istanbul, Turkey
{opinarer,aozgovde}@gsu.edu.tr

Abstract. High energy consumption of sensor devices is a major problem in smart building systems, since it strongly impacts the system lifetime. However, existing approaches are often fitted to a single monitoring application and rely on static configurations for sensor devices: optimization of their acquisition and transmission frequencies to actual multiple application requirements is not tackled. In this paper, we focus on energy-aware dynamic sensor device re-configuration to lower energy consumption while fulfilling real-time application requirements. We introduce the Smart-Service Stream-oriented Sensor Management (3SoSM) that binds together sensor configuration and management of sensor data streams. We present a multi-application monitoring system architecture that optimizes application requirements for data streams into sensor device configurations, and we relate the experiments with our experimental platform.

Keywords: Sensor data management · Smart building · Wireless sensor network · Continuous query processing

1 Introduction

Nowadays it is well known that traditional buildings are primary consumers of a significant portion of energy resources, thus, most of the world's cities are concerned by the potential of smart buildings. The design of these buildings is based on sustainable construction standards to consume less energy than traditional buildings and to minimize their impacts on the natural environment. Smart building technology brings in some nice features such as security, comfort and accessibility, however with extra constraints to acquire and analyze "Big" and/or "Fast" Data generated by devices like sensors: providing useful services for occupants such as thermal comfort, air quality, physical security, etc., comes at the cost of managing/processing large and complex real-time datasets. Big Data, with its 5 Vs (Volume, Velocity, Variety, Veracity, Value), represents a new

This work is supported by INSA-Lyon, LIRIS Research Laboratory.

era in data exploration and utilization, and tackles sensing, analysing, sharing, storing, querying, etc., very large amount of data [1].

Smart building technology can enhance the energy consumption of buildings, however its infrastructure often consists of a wireless sensor network with devices that have limited energy and battery lifetime [2]. These devices are autonomous in terms of energy: their energy consumption determines their lifespan. In general, sensor devices periodically sample physical quantity measures and transmit them with a defined frequency to fulfill requirements of smart building applications. Tougher requirements (more measures and/or faster frequencies) will inevitably lead to a greater energy consumption for these devices.

In this study, we focus on a sustainable architecture for multi-application monitoring systems that continuously adapt to application requirements, context and user configuration. We consider a monitoring system as a set of applications that exploit sensor measures in real-time, where these applications are declaratively expressed as (service-oriented) continuous queries over sensor data streams. This architecture supports multiple applications in parallel, with dynamic requirements: it can gracefully handle several different requests for a same device. However, when using a static configuration for acquisition and transmission frequencies of devices, energy consumption of the monitoring system can not be optimized with regards to actual application requirements.

In this paper, we propose a dynamic sensor configuration mechanism to avoid unnecessary data measurements and to promote less expensive data transmission for sensor devices. We present ***S****mart-* ***S****ervice* ***S****tream-****o****riented* ***S****ensor* ***M****anagement* (***3SoSM***), an approach to optimize interactions between application requirements and Wireless Sensor Network (*WSN*) environment in real-time at the gateway level, independently of inner application logic. Our *3SoSM* approach performs energy-aware dynamic sensor device re-configuration to lower energy consumption while fulfilling real-time application requirements.

The remainder of this paper is organised as follows: Related works are given in Sect. 2. Section 3 presents an overview of our multi-application monitoring system architecture. Optimization of application requirements into sensor device configurations is explained in Sect. 4. Section 5 gives a brief description of our experimental platform to implement our approach and Sect. 6 describes the experiments we conducted. Finally, conclusions are presented in Sect. 7.

2 Related Works

Smart buildings are an application domain for the more general notion of pervasive environments, that consist of physical devices, wireless sensors, actuators, middlewares, applications, etc. These components and platforms compose the infrastructure of a building management system. Along with *WSN*, other components are covered by different research domains: smart building design, sensor data management, energy management, etc.

In the literature, existing studies mainly focus on design and data management sides. [3] proposes a model for monitoring systems: a real-time decision unit that interacts with sensors for diagnosis of the building's state and with the building's controllers to select the appropriate interventions. [4] introduces a smart home energy control system and a smart interface to provide services to occupants. The authors focus on especially on lighting systems to reduce cost. [5] presents an occupant-centric design based on gathering/visualization of high density sensor network dataset to make the pervasive environment sustainable. [6] proposes a model for a better understanding of urban phenomena by exploring and exploiting heterogeneous data from various sources, such as physical sensors, surveys, social networks, etc.

Besides, energy consumption of building system is also discussed. For instance, [7] proposes an energy management technique to handle computational needs of ambient intelligence applications by using energy minimization workload assignment policies. [8] introduces the necessity of having a monitoring/control system for a building and proposes deploying digital smart meters that communicate wirelessly. The main idea is to search which equipments and system characteristics are responsible for the energy consumption of the smart meters.

[9, 10] are the closest studies to ours. [9] presents intelligent building architecture based on self-adapting intelligent gateway. This gateway handles service decisions, device management, data aggregation, occupant-based pattern generation and provision of energy management services. A novel self-adapting intelligent system is introduced and proposed system can save approximately 16–24% energy. [10] presents self-adapting algorithms for context-aware systems. Proposed approach detects and analyses changes in the environment, decides how system should react respecting the given set of policies. In that approach, deployed sensor devices and actuators are capable to take adapted decisions. Approaches of [9, 10] have some common characteristics: to propose a dynamic management system for smart building environment while processing user preferences. However, these studies are bounded by predefined building applications and application requirement-sensor configuration relation is not established. Besides, their approaches do not benefit from potential reconfiguration of acquisition and transmission frequencies: sensor configuration stays static during the system lifetime. Moreover, high energy consumption is not considered as a major issue. In our *3SoSM* approach, we introduce an energy-aware dynamic sensor reconfiguration process while fully fulfilling application requirements.

3 Overview of Multi-application Monitoring System

Smart building management systems are one of the main application area of pervasive environment research domain. Smart building systems are composed by wireless sensor devices, hence, high energy consumption and limited service lifetime are crucial problems. In this study, we focus on the energy consumption of monitoring architectures supporting multiple smart building applications.

▷ ***Monitoring Architecture for Smart Building Applications:*** In this study, we adopt a "declarative monitoring architecture", build upon a pervasive environment management system (PEMS) using declarative (SQL-like) continuous queries that can interact with distributed devices like sensors. This architecture has 3 main layers, as illustrated in Fig. 1:

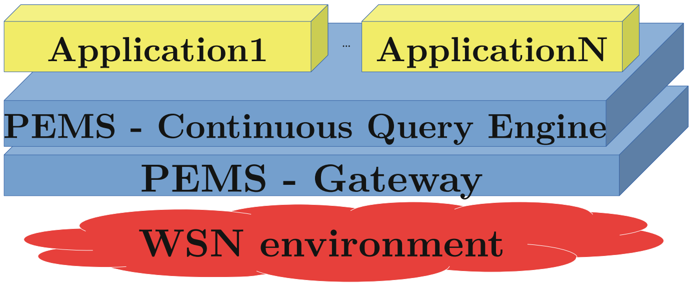

Fig. 1. Declarative monitoring architecture

- *Application layer*, where application requirements are defined, declaratively expressed as a set of continuous queries over distributed services [11];
- *PEMS* that integrates non-conventional, dynamic and heterogeneous data sources and manages query executions; it includes a continuous query engine that interacts with services provided by the lower layer through the logical gateway, and manages queries coming from application layer [11,12];
- *WSN environment*, where wireless sensor devices acquire physical quantity measures, and can communicate with other sensor devices and physical gateways.

▷ ***Pervasive Environment Management Framework:*** We adopt the *SoCQ* (**S**ervice-**o**riented **C**ontinuous **Q**uery) framework [11] for the *PEMS* layer. It takes a data-oriented perspective on pervasive environments such as smart buildings. It provides a unified view and access to various and heterogeneous data resources, or services, available in the environment. *XD-relations* (eXtended Dynamic Relations) can represent standard relations, that may be updated, or data streams, that continuously produces data. Pervasive applications can then be created in a declarative fashion using service-oriented continuous queries over *XD-relations*. Queries may be one-shot queries (like standard SQL queries) or continuous queries (with a dynamic result, like a stream). Queries can also interact with distributed services: service discovery, method invocation, stream subscription. Furthermore, invocations and subscriptions can be finely parametrized.

For instance, a service discovery query can search for sensor services that provide a location, a method to get the current temperature, and a continuous stream of temperatures. The result is a XD-relation with a ServiceID, a Location, and a virtual attribute for Temperature. Once relevant services are listed, a continuous query can subscribe to the temperature stream of every discovered services, to build a resulting data stream with Temperature values. If new services are discovered and/or some services become unavailable, the continuous query automatically adapts the set of stream subscriptions.

▷ ***Smart Service-Stream Oriented Sensor Management - 3SoSM:*** *SoCQ* handles a multi-application mechanism and supports multiple parametrized subscriptions to the same service. Moreover, it supports real-time user configuration of applications and context-aware applications through queries that can dynamically combine data, streams and services. Interactions with services (discovery, invocations, subscriptions) are handled by a gateway positioned between the *PEMS* layer and the *WSN* environment. However, *SoCQ* does not tackle the issue of sensor configuration, and, thus, adopt a static configuration for sensor devices. Like for other approaches presented in the Related Works section, it is an issue for the energy enhancement of the system. Based on the existence of **dynamically configurable wireless sensor devices** [13], we propose a novel approach: a energy-aware dynamic sensor configuration based on real-time application requirements to improve energy consumption of the system.

From our perspective, application requirements are introduced by applications configured by users, as a set of continuous queries. For instance: "*Application A1 computes the average temperature over the last 10* min*, with an update every 5* min*, with an accuracy of 1* s *and a maximum latency of 1* min". Those application requirements in terms of data management can be summarized by the following parameters:

- ***temporal window size*** introduces the time interval for calculating the result;
- ***periodicity of result update*** stands for the refreshing rate of the result;
- ***data acquisition periodicity*** represents the temporal accuracy of measure;
- ***maximum latency*** presents the maximum acceptable delay between the acquisition of data and its transmission to the PEMS layer for result calculation.

The first two parameters concern the computing of the result, whereas *data acquisition periodicity* and *maximum latency* parameters are related to sensor devices and acquisition/transmission of data. In this study, we propose ***S****mart* ***S****ervice-****S****tream* ***O****riented* ***S****ensor* ***M****anagement (3SoSM)*: the main principle is to compute a data acquisition/transmission schedule for each device, called **sensor configuration oriented pattern** or SCO-pattern, based on application requirements at a given time instant. As application requirements change over time, *3SoSM* dynamically reconfigures sensor devices by updating their specific SCO-patterns, so as to avoid unnecessary data measurements and to enable aggregated data transmission (less and/or smaller network packets to be sent).

In summary, proposed approaches in the literature adopt static sensor configuration and are specialized for specific applications. However, existence of devices with a dynamic sensor configuration feature offers a different perspective. In this context, we propose the *3SoSM* approach that provides finer sensor configuration than duty-cycle and similar techniques. Our proposition of dynamic sensor management based on real-time application requirements is performed at the gateway layer, in order to optimize energy consumption of sensor devices independently from the application layer and/or the query engine.

4 Optimization of Application Requirements

In this section, we present the core of our *3SoSM* approach: an algorithm to compute data acquisition/transmission schedule for each sensor devices, based on real-time application requirements. We first present a formalization of application requirements (subscription requests) and of sensor configuration (SCO-patterns), and then the algorithm itself (called GeNoMe process).

4.1 Formalization of Application Requirements and Sensor Configuration

$\triangleright$ ***Query requirements:*** A typical smart building is equipped with various wireless sensor devices $d_i \in D$, where D is the set of devices in the environment. Each sensor device may have multiple functionalities to acquire physical quantity measures $m_i \in M$, e.g., temperature, humidity. In our *3SoSM* approach, application requirements are defined within queries in terms of data source requirements (targeted sensors d and measures m) and of temporal requirements: temporal window size β, periodicity of result updates p^{upd}, measure acquisition periodicity p^{acq} and maximum latency of measure transmission *latency*. The unit for all these temporal parameters is the second (or millisecond, if required). Definition of these parameters have been introduced in the previous section.

We then represent application requirements on sensors at a given time instant by a set of parametrized subscription requests $\{s_1, s_2, \ldots, s_n\}$, where:
$s_i = (d_i, m_i, \beta_i, p_i^{upd}, p_i^{acq}, latency_i) \in D \times M \times \mathbb{N}^* \times \mathbb{N}^* \times \mathbb{N}^* \times \mathbb{N}$

$\triangleright$ ***SCO-Patterns:*** We propose a ***sensor configuration oriented pattern***, or SCO-pattern, to represent a data acquisition/transmission schedule used to configure a physical device. A SCO-pattern consists of a list of sensor events and the length ℓ of this pattern. A sensor event is a $< timestamp, action >$ couple. Here, we define two types of event actions: either a ***data acquisition A*** or a ***data acquisition and transmission AT***. Event timestamps are enclosed by time interval $]0;\ell]$. The length of the pattern introduces the periodicity of the pattern: a sensor executes this pattern repetitively every ℓ seconds. Thus, a SCO-pattern P is denoted by:
$\mathrm{P} = (\{(t_i, a_i)\}, \ell)$ with $\ell \in \mathbb{N}^+$, $t_i \in]0;\ell]$, $a_i \in \{A, AT\}$.

For instance, the SCO-pattern for a sensor device that should measure every second and transmit every 3 s is: $\mathrm{P} = (\{(1, A), (2, A), (3, AT)\}, 3)$.

$\triangleright$ ***Algorithm and intermediate DOA-Patterns:*** We design an algorithm to optimize a set of subscription requests for a sensor device into a SCO-pattern defining the configuration of this device. This algorithm, presented in the next section, relies on an intermediate ***data-oriented acquisition pattern***, or DOA-pattern, that can represent a single subscription request and can also be merged.

In a similar way to SCO-patterns, a DOA-pattern consists of a list of acquistion-latency events and the length ℓ of this pattern. An acquistion-latency event is a triple $< timestamp, action, latency >$, where action is always an acquisition. A DOA-pattern ρ is denoted by:
$\rho = (\{(t_i, A, latency_i)\}, \ell)$, with $\ell \in \mathbb{N}^+$, $t_i \in]0;\ell]$.

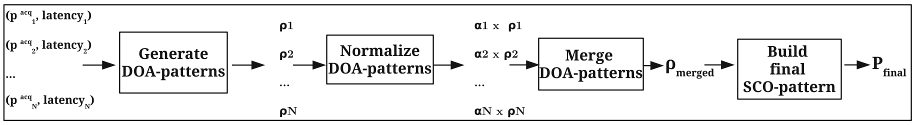

Fig. 2. Overview of the GeNoMe process

For instance, the DOA-pattern for a subscription request that requires a measure every second and a maximum latency of 3 s is: $\rho = (\{(1, A, 3)\}, 1)$. Here, a single acquisition-latency event is sufficient to describe this request.

4.2 Algorithm

The goal of this algorithm is to generate a sensor device configuration, or SCO-pattern, that fulfills all the application requirements expressed by a set of subscription requests targeting this device (with their specific acquisition period and latency). Even though a sensor device can have multiple functionalities such as measuring temperature and humidity, we here consider subscription requests only for a single physical measure (multi-modality patterns is a work in progress).

The generation process of the final SCO-pattern P_{final} is named *3SoSM GeNoMe* process (***Ge****nerate*-***No****rmalize*-***Me****rge*) and illustrated in Fig. 2. We now detail the 4 steps of this algorithm (due to space restrictions, we do not include the listings of pseudo-code for these steps):

1. ***Generate DOA-Patterns:*** we generate a DOA-pattern ρ_i for each subscription s_i with its parameters $(p^{acq}, latency)$: $\rho_i = (\{(p^{acq}, A, latency)\}, p^{acq})$.
2. ***Normalize DOA-Patterns:*** In order to merge DOA-patterns (next step), they should all have the same length, that will be the length of the final pattern. The lowest common multiple method is used on pattern lengths (in fact, acquisition periods): $\ell_{final} = LCM(\ell_i, \ell_j, \ldots) = LCM(p_i^{acq}, p_j^{acq}, \ldots)$. Then, a coefficient for each pattern is calculated: $\alpha_i = \ell_{final}/\ell_i \in \mathbb{N}$. This coefficient indicates how many times each pattern should be repeated in order to reach the length of the final pattern, i.e., to normalize this pattern. Thus, each pattern is extended with its specific coefficient: $\alpha_i \times \rho_i$.
3. ***Merge DOA-Patterns:*** We now merge the set of DOA-patterns to obtain the merged DOA-pattern ρ_{merged}, with the normalized length. The lists of events are merged into a single list. If two events occur at the same timestamp, those events are themselves merged into a single event with a "merged" latency (the minimum latency of those two events). Event actions are still always acquisition actions (no transmission action).
4. ***Build final SCO-Pattern:*** The DOA-pattern ρ_{merged} indicates data acquisition timestamps and latency, and the periodicity of that pattern. Data transmission is not indicated yet. As a final step, transmission actions are inserted on some events and latency values are then removed, in order to build a final SCO-pattern. Optimal transmission events are calculated based on latency

values of acquisition events, to fulfil maximum latency requirement for each acquisition event. As a first heuristic, starting from the first acquisition event, we search for the latest next event whose measure can be transmitted with all previous measures while respecting latency constraints. We then continue with the next acquisition event, until the end of the pattern.

Example: Suppose that there are two subscription requests to the same sensor device d_i for measuring temperature M_T. The first subscription requires data acquisition every *4* s with latency *7* s ($p^{acq} = 4\,\mathrm{s}, latency = 7\,\mathrm{s}$). The second subscription requires data acquisition every *5* s with latency *9* s ($p^{acq} = 5\,\mathrm{s}, latency = 9\,\mathrm{s}$). Regardless of window sizes β and result update periods p^{upd}, those subscriptions can be expressed as:
$s_1 = (d_i, M_T, \beta_i, p_i^{upd}, 4, 7)$ and $s_2 = (d_i, M_T, \beta_j, p_j^{upd}, 5, 9)$.

With our algorithm, we optimize this set of subscriptions into a SCO-pattern:

1. ***Generate DOA-Patterns:*** Subscription requests can be expressed as:
$\rho_1 = (\{4, A, 7\}, 4)$ and $\rho_2 = (\{5, A, 9\}, 5)$
2. ***Normalize DOA-Patterns:*** Normalized length is LCM(4,5)=20. Coefficients are $\alpha_1 = 20/4 = 5$ for ρ_1 and $\alpha_2 = 20/5 = 4$ for ρ_2. Then, ρ_1 should be repeated five times and ρ_2 should be repeated four times:
$\alpha_1 \times \rho_1 = 5 \times \rho_1 = (\{(4, A, 7), (8, A, 7), (12, A, 7), (16, A, 7), (20, A, 7)\}, 20)$
$\alpha_2 \times \rho_2 = 4 \times \rho_2 = (\{(5, A, 9), (10, A, 9), (15, A, 9), (20, A, 9)\}, 20)$
3. ***Merge DOA-Patterns:*** The merged pattern has a length of 20, and only the last event (at 20) merges two events (with a latency of MIN(7,9)=7):
$\rho_{merged} = (\{(4, A, 7), (5, A, 9), (8, A, 7), (10, A, 9), (12, A, 7), (15, A, 9),$
$(16, A, 7), (20, A, 7)\}, 20)$
4. ***Build final SCO-Pattern:*** Transmission actions are added to relevant events at 10, 16 and 20 – all latency requirements are thus fulfilled:
$\mathrm{P}_f = (\{(4, A), (5, A), (8, A), (10, AT), (12, A), (15, A), (16, AT), (20, AT)\}, 20)$

We remark that the final pattern requires only 18 transmission actions per 2-min (3 AT per 20 s), whereas the two intial requests would require respectively 15 and 12, and a total of 24 AT considering a common AT every 40 s.

5 Experimental Platform

5.1 Continuous Query Engine: SoCQ Engine

In this study, we use the *SoCQ Engine* [11]. Benefits of the *SoCQ* framework as a pervasive environment management system was already introduced in Sect. 3. *SoCQ Engine* is a service-oriented continuous query engine implemented in Java. A Data Description Language (DDL) allows to define *XD-Relations*, and a SQL-like query language allows to specify one-shot and continuous queries over *XD-Relations* [12]. A user interface controls the query engine: users can visualize *XD-Relations* and their content, and launch one-shot/continuous queries.

5.2 WSN Simulator: Modified WSNet

In our platform, we integrated the *WSN* simulator *WSNet* [14]. *WSNet* is a modular event-driven simulator, more precisely a discrete event simulator (DES). *WSNet* adopts basic functionality of DES: in order to avoid simulating every time splice, the time line is split into events and no change is presumed to occur in the system between consecutive events; thus the simulation can directly jump in time from one event to the next. However, the *PEMS* works on real-time. To avoid time scheduling difference, **we modified the time scheduler** of the *WSNet* simulator: we introduced a time factor to run experiments in real-time or n-times faster ($\times$10, $\times$20...).

5.3 Gateway: 3SoSM Gateway

In the *3SoSM* architecture (Fig. 1), the *Gateway* is a technical bridge between two environments: it manages interactions and bidirectional communication between the *PEMS* and the *WSN*. We implemented the *3SoSM* principles in a ***3SoSM Gateway***. It is implemented in Java, and interacts with the *SoCQ Engine* and *WSNet*. *3SoSM Gateway* has two primary modules: the ***Service Manager***, that manages *SoCQ* services representing available sensor devices; and the ***Subscription Manager***, that continuously analyzes application requirements to generate new SCO-patterns for sensor devices when required.

6 Experiments

6.1 Experiment Setup

The simulations are performed on the *modified WSNet*. We simulate one part of the topology of our physical platform *SoCQ4Home* deployed in our research laboratory [15]: 70 simulated sensor devices are located at specific positions over a floor of the building (10 m × 60 m × 4 m). The deployed sensor devices have fixed positions during the simulation and we consider that they have enough energy until the end of the simulation. We adopt most known pervasive environment communication protocol *Zigbee IEEE 802.15.4*, simulated with a UDG propagation model and 35 m transmission range, and basic radio module states for devices. Calculation of energy consumption is based on CPU and radio components, adopted from [16,17].

6.2 Experimental Scenario

In a typical scenario, multiple applications launch continuous queries concerning sensors to the *PEMS*. The gateway manages required parametrized stream requests and, with *3SoSM*, generates optimal sensor configurations. To evaluate our approach, a scenario is performed with and without using *3SoSM* during one day (1440 min) to observe concrete performance of data stream management and evolution of energy consumptions. We designed two scenarios: "Comfort Temperature Range" and "Temperature of Occupied Rooms". In this paper, we present only one of the performed scenarios.

Scenario "Comfort Temperature Range": Smart buildings are responsible from thermal comfort of occupants. Based on the outdoor temperature, an indoor comfort temperature range is determined. During this scenario, there are two subscription requests to each sensor device. A first application requires temperature of each room with a data acquisition every 15 s and a latency of 60 s. A second application has a specific condition: it demands to track more frequently temperature of rooms that are out of the current comfort temperature range, with a data acquisition every 1 s and a latency of 4 s. A set of 3 *SoCQ* queries are implemented: 1 discovery query to determine available sensor devices, and 1 stream query for each application. The subscription requests to sensor services are generated by the *PEMS* and, based on these parametrized subscriptions, dynamic sensor configuration is realized by the *3SoSM Gateway.*

The first application receives a temperature data stream from each sensor device, however the second application receives data streams only from sensors in rooms where temperature is out of the current comfort range. Users want to track temperature of these rooms more frequently until temperature is in the comfort range again. Sensors are dynamically re-configured by our *3SoSM* Gateway when required. Since data transmission is the most expensive action on sensor side (in terms of energy consumption), we expect a higher energy consumption for sensor devices that are in a room where temperature is not in the comfort range. This functionality shows the context-aware feature of *3SoSM* approach. For the evaluation of our approach, we compare the results of dynamic re-configuration by *3SoSM* Gateway with the results of a static configuration of sensors that always fulfills requirements of both applications: $p^{acq} = 1\,\text{s}$, $latency = 4\,\text{s}$.

Experiment Result: Figure 3 shows the evolution of energy consumption of a single wireless temperature sensor in a room during the simulation with and without dynamic re-configuration. Upper graph presents indoor temperatures and dynamic comfort temperature ranges (dashed curves). Regions where indoor temperature is outside of comfort temperature range are visible. Lower graph shows the decrease of energy level, and thus lifetime, of that sensor device. Energy consumption increases while room temperature is outside of comfort temperature range: the second subscription is then requested, hence, data transmission frequency is increased from 60 s to 4 s.

Results show that the energy consumption of that sensor device is higher without dynamic re-configuration: as a consequence, the sensor device dies earlier. With a static configuration, it nearly dies at the end of the day, whereas it still has energy with dynamic re-configuration. Initial energy level of sensor device is here set on purpose to emphasize the lifetime difference between both cases. Based on the given scenario parameters, we achieve to reduce additional communication cost: For instance, while temperature is inside the given comfort range, without our approach, a single temperature sensor sends 900 data packets to base station during one hour. With the *3SoSM* Gateway, number of

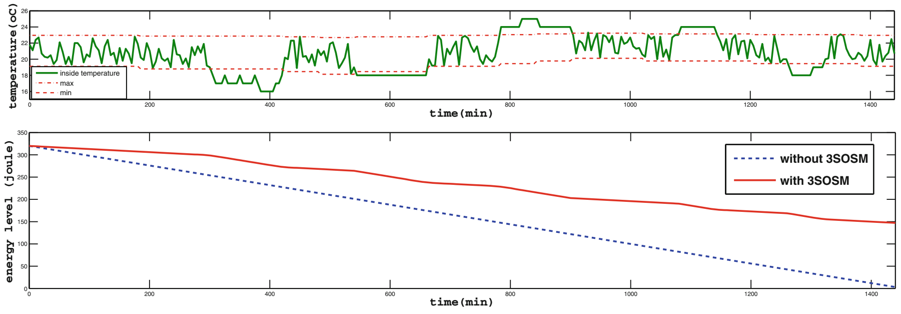

Fig. 3. Indoor temperature variations and energy level evolution of a sensor device with static configuration or dynamic re-configuration (1440 min, 1 day)

transmitted packet is only 60 for the same time period. Thus, at the end of the experiment, 83 % of energy level is saved by using the *3SoSM* approach.

Our approach can increase the lifetime of a sensor device. Let's consider a basic 9 V alkaline battery ($\simeq$ 18.8kJ) connected to our sensor device. With given parameters, the sensor can survive only 59.5 days with a static configuration, however its lifetime can be extended up to 109.08 days with dynamic re-configuration. Here, *3SoSM* lengthens sensor lifetime by $\simeq$ 83 %. Obviously, service lifespan and the energy savings depend on the application requirements and context (here, actual temperature).

7 Conclusion

In this paper, we point at a major challenge of smart building technology: energy consumption of the monitoring architecture itself while dealing with data exploration/utilization. We expose that existing studies do not tackle neither energy consumption of deployed equipments nor lifetime of the whole system. Existing approaches commonly adopt static configurations for wireless devices. Here, we focus on the lifetime of a monitoring system and introduce new mechanisms for dynamic reconfiguration of sensor data acquisition/transmission schedules.

We present a sustainable declarative monitoring architecture to process massive raw data from sensors. We rely on declarative *PEMS* principles and tackle the energy optimisation of interactions between application real-time requirements and sensor devices. We introduce our approach *3SoSM* based on data-driven acquisition and transmission time patterns. We propose the *GeNoMe-X* process to optimize multiple parametrized subscriptions to a same device. We implemented a *3SoSM* Gateway that supports the *GeNoMe-X* process to fulfil dynamic application requirements. We conducted experiments using the *SoCQ* engine and a modified *WSNet* simulator. Impacts of our approach on energy consumption and on lifetime are presented and discussed.

As perspectives for *3SoSM* approach, we plan to extend time patterns to support multi-modality, and to benefit from real raw data measured by physical

SoCQ4Home platform. Besides, we also plan to integrate energy-aware dynamic sleep scheduling mechanism based on predictions, as a complementary of data driven scheduling. Furthermore, we are studying ways to decentralize parts of the optimization process into the *WSN*, on the smart devices themselves.

References

1. Zikopoulos, P., Eaton, C., et al.: Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. McGraw-Hill Osborne Media, New York (2011)
2. Aloui, I., Kazar, O., Kahloul, L., Servigne, S.: A new itinerary planning approach among multiple mobile agents in wireless sensor networks (wsn) to reduce energy consumption. IJCNIS 7(2) (2015)
3. Doukas, H., Patlitzianas, K.D., Iatropoulos, K., Psarras, J.: Intelligent building energy management system using rule sets. J. Building Environ. **42**(10), 3562–3569 (2007)
4. Han, D.-M., Lim, J.-H.: Smart home energy management system using ieee 802.15.4 and zigbee. IEEE Trans. Consum. Electron. **56**(3), 1403–1410 (2010)
5. Khan, A., Hornbæk, K.A.S.: Big data from the built environment. In: Proceedings of the 2nd International Workshop LARGE 11, pp. 29–32. ACM (2011)
6. Sylvie, S., Gripay, Y., Jean-Michel, D., Céline, N., Jacques, J., Olivier, C., Radouane, M.: Data science approach for a cross-disciplinary understanding of urban phenomena: Application to energy efficiency of buildings. Procedia Eng. **115**, 45–52 (2015)
7. Zapater, M., Sanchez, C., Ayala, J.L., Moya, J.M., Risco-Martın, J.L.: Ubiquitous green computing techniques for high demand applications in smart environments. Sensors **12**, 10659–10677 (2012)
8. Preisel, M., Diaz, A., Wimmer, W.: Energy consumption of smart meters. J. Inf. Commun. Technol., 37 (2013)
9. Byun, J., Park, S.: Development of a self-adapting intelligent system for building energy saving and context-aware smart services. IEEE Trans. Consum. Electron. **57**(1), 90–98 (2011)
10. Cioara, T., Anghel, I., Salomie, I., Dinsoreanu, M., Copil, G., Moldovan, D.: A self-adapting algorithm for context aware systems. In: RoEduNet 2010, pp. 374–379. IEEE (2010)
11. Gripay, Y., Laforest, F., Petit, J.-M.: Socq: A framework for pervasive environments. In: ISPAN 2009, pp. 154–159. IEEE (2009)
12. Gripay, Y., Laforest, F., Petit, J.-M.: A simple (yet powerful) algebra for pervasive environments. In: EDBT 2010, pp. 359–370. ACM (2010)
13. Perera, C., Zaslavsky, A., Compton, M., Christen, P., Georgakopoulos, D.: Semantic-driven configuration of internet of things middleware. In: SKG 2013, pp. 66–73. IEEE (2013)
14. Wsnet/worldsens simulator. http://wsnet.gforge.inria.fr/. Accessed: 22 Feb 2016
15. Socq4home project. http://liris.cnrs.fr/socq4home/. Accessed: 04 Feb 2016
16. Pinarer, O., Ozgovde, A.: Improving the energy efficiency of wearable computing units using on sensor fifo memory. Int. J. e-Education, e-Business, e-Management e-Learning **5**(2), 105 (2015)
17. Pinarer, O., Ozgovde, A.: Application specific dynamic sleep scheduling. In: 23rd Signal Processing and Communications Applications Conference, SIU 2015, pp. 1765–1768. IEEE (2015)