



## Dynamic energy-aware sensor configuration in multi-application monitoring systems

Ozgun Pinarer, Yann Gripay, Sylvie Servigne, Ozgovde Atay, Atilla Baskurt

### ► To cite this version:

Ozgun Pinarer, Yann Gripay, Sylvie Servigne, Ozgovde Atay, Atilla Baskurt. Dynamic energy-aware sensor configuration in multi-application monitoring systems. *Pervasive and Mobile Computing*, 2017, 41, pp.192-204. 10.1016/j.pmcj.2017.08.005 . hal-01585263

**HAL Id: hal-01585263**

**<https://hal.science/hal-01585263>**

Submitted on 12 Sep 2017

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Dynamic Energy-Aware Sensor Configuration in Multi-Application Monitoring Systems

Ozgun Pinarer<sup>a,b</sup>, Yann Gripay<sup>a</sup>, Sylvie Servigne<sup>a</sup>, Atay Ozgovde<sup>b</sup>, Atilla Baskurt<sup>a</sup>

<sup>a</sup> *Univ Lyon, CNRS, France*  
*INSA-Lyon, LIRIS, UMR5205, F-69621*  
*Email: name.surname@insa-lyon.fr*

<sup>b</sup> *Galatasaray University*  
*Ciragan Cad. No:36 34349 Istanbul, Turkey*  
*Email: {opinarer, aozgovde}@gsu.edu.tr*

---

## Abstract

A typical pervasive monitoring system like a smart building depends on an infrastructure composed of hundreds of heterogeneous wireless sensor devices. Managing the energy consumption of these devices poses a challenging problem that affects the overall efficiency and usability. Existing approaches for sensor energy consumption typically assume a single monitoring application to consume sensor data and a static configuration for sensor devices. In this paper, we focus on a multi-application context with dynamic requirements and multi-modal sensor devices. We present 3SoSM, an approach to optimize interactions between application requirements and wireless sensor environment in real-time. It relies on an energy-aware dynamic configuration of sensor devices to lower energy consumption while fulfilling application requirements. To bind together sensor configuration and dynamic management of data streams, we design a sustainable multi-application monitoring system architecture for pervasive environments that collects application requirements for sensor data streams and optimizes them into sensor configurations. To demonstrate the effectiveness of our approach, a set of experiments are designed in the context of smart buildings. We comparatively evaluate our approach to show how dynamic sensor configuration for multiple monitoring applications indeed outperforms the mainstream duty-cycling method.

*Keywords:* pervasive environment, sensor data management, continuous query processing, wireless sensor network

---

## 1. Introduction

Smart building applications have long gained attention in the scientific community and nowadays in the industry. Buildings, whether smart or not, are among the primary consumers of the available energy sources today. Therefore, considerable effort is poured into the design of smart buildings where energy consumption is tried to get decreased without sacrificing the satisfaction of the occupants. To accomplish this aim, the components of the building itself, as well as the context of its users, should be continuously monitored by means of streams of data from sensory inputs.

A typical pervasive monitoring system like a smart building consists of multi-modal wireless sensor devices that are equipped with sensing, processing and communication facilities. Sensing part can measure physical quantities about the environment with some given sampling rate (sampling frequency), processing part is able to do some computation on the measured values and communication part is able to listen and send data packets to other sensor devices. These deployed wireless devices are autonomous in terms of energy: they have limited energy and battery lifetime. Smart building technology mainly relies on such wireless sensing infrastructure. In this field, initial commercial solutions are **Building Automation Systems** where a specific set of functionalities such as heating and ventilation control and lighting management is implemented by a single vendor. However, as the sensor infrastructure is becoming a standard component incorporated into the design and construction of the modern buildings, flexible application development by a third party is becoming an issue rather than the infrastructure itself.

In this context, one of the main contributions of this paper is to allow multiple applications to exploit the same smart building infrastructure while considering energy consumption of the sensor devices in this infrastructure. Our energy-aware monitoring system continuously adapt to various application requirements, to building actual context and to user configuration. These applications operate on a network of multi-modal wireless sensor devices and use declarative continuous queries to exploit sensor data streams. Application requirements as declared by the application developer can be fulfilled in a multitude of ways: our method translates these requirements as energy efficient acquisition and transmission schedules for the wireless sensor

network. We propose *Smart-Service Stream-oriented Sensor Management* (3SoSM), an approach to optimize interactions between application requirements and the wireless sensor environment in real-time. The core of our approach is the definition of a Schedule Time Pattern with acquisition, transmission and reception actions for each device configuration. The dynamic reconfiguration of devices is performed through the real-time update and optimization of Schedule Time Patterns according to application requirements that may change over time. With this approach, we expect to avoid unnecessary data measurements that may occur with static configuration and to promote grouped or even compressed data transmission when possible.

In this article, Section 2 presents an overview of our multi-application monitoring system architecture. Formalization of our approach is explained in Section 3 and the energy-aware optimization process is detailed in Section 4. Section 5 gives a brief description of our experimental platform to implement our approach and Section 6 describes the experiments we conducted. Experiment results are discussed in Section 7. Related works are given in Section 8 and finally, conclusions are given in Section 9.

## 2. Overview of 3SoSM

Our 3SoSM approach focuses on a multi-application monitoring system with multi-modal sensor devices (i.e. devices that can measure different physical quantities) and 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 level, in order to optimize energy consumption of sensor devices independently from the application layer and/or the query engine.

### 2.1. Monitoring Architecture for Smart Building Applications

In this study, we adopt a “declarative monitoring architecture”, built upon a **Pervasive Environment Management System (PEMS)** as presented in [1, 2, 3]. Using declarative (*SQL-like*) continuous queries, an application can easily interact with distributed devices like sensors. Figure 1 presents our declarative monitoring application that has 4 main layers: Application, PEMS Query Engine, PEMS Gateway and WSN (Wireless Sensor Network). **Application** layer provides end-user access to the monitoring system. Application requirements are defined at this layer and are declaratively expressed as a set of continuous queries over distributed services [1]. **PEMS**

Query Engine is responsible for managing query executions of queries coming from the Application layer. This layer integrates sensor devices as non-conventional, dynamic and heterogeneous data sources. To manage that, it includes a continuous query engine that interacts with services provided by the environment, in particular sensor services. PEMS Gateway stands for managing bidirectional communication and interactions between PEMS Query Engine and WSN. WSN layer represents wireless sensor devices that acquire physical quantity measures and can communicate with other sensor devices and physical gateways.

We use the Service-Oriented Continuous

Query (SoCQ) framework [1] for the PEMS Query Engine. The SoCQ engine handles a multi-application mechanism and supports multiple parameterized stream subscriptions to the same device. Moreover, it supports real-time user configuration of applications and context-aware applications through queries that can dynamically combine data, streams and services. The user

may insert complex continuous queries with conditions over data streams to trigger new interactions with services during their execution. Interactions with services (discovery, invocations of methods, subscriptions to data streams) are handled by PEMS Gateway. However, the SoCQ Engine itself does not overcome the reconfiguration of sensor devices, and thus assumes a pre-existing static configuration. Like for other approaches presented in the Related Works section, it is an issue for the energy enhancement of the system.

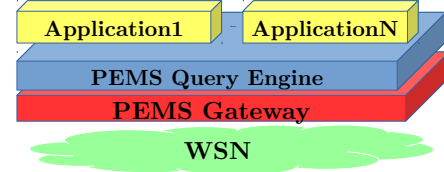


Figure 1: Declarative Monitoring Architecture.

## 2.2. From Dynamic Application Requirements to Dynamic Sensor Configurations

We consider that applications define their requirements as a set of continuous queries over parameterized data streams produced by sensor devices. For instance: “*Application A1 computes the average temperature over the last 10 minutes, with an update every 5 minutes, with an accuracy of 1 second and a maximum latency of 1 minute*”. In this example, the application requests temperature measure data streams from all devices with temperature sensors. To express the application requirements concerning sensor data management, we consider the following parameters targeting sensor measures: • **temporal**

108 **window** introduces the time interval for calculating the result (for the given  
 109 example: 10 min); • **update periodicity** stands for the refreshing rate of  
 110 the result (5 min); • **acquisition periodicity** represents the temporal  
 111 accuracy of measures (1 sec); • **maximum latency** presents the maximum  
 112 acceptable delay between the acquisition of data and its transmission to the  
 113 PEMS layer for result calculation (1 min). **temporal window** and **update**  
 114 **periodicity** concern the computing of the result, whereas **acquisition**  
 115 **periodicity** and **maximum latency** are related to acquisition/transmission  
 116 of data by sensor devices.

117 Here, we propose a novel approach: management and optimization of the  
 118 real-time application requirements to enhance the energy consumption of the  
 119 whole system. To fulfill the application requirements, we define a Schedule  
 120 Time Pattern for each sensor device and configure devices accordingly. Our  
 121 approach assumes that sensor devices are dynamically configurable [4].

### 122 3. 3SoSM Formalization

123 We consider that application requirements are parameterized stream sub-  
 124 scription requests and the network topology is a tree topology. A solution  
 125 to our optimization problem is individual Schedule Time Patterns, namely  
 126 SCO-PATTERNS.

#### 127 3.1. Application Requirements

128 Let  $D$  be the set of wireless sensor devices in the WSN environment. Each  
 129 sensor device  $d_i \in D$  may have multiple modalities to acquire different phys-  
 130 ical quantity measures  $m_i \in M$ , e.g., temperature, humidity. Application  
 131 requirements are defined in terms of data source requirements: targeted sen-  
 132 sors  $d$  and requested measures  $m$ ; and of temporal requirements: temporal  
 133 window  $\beta$ , update periodicity  $p^{upd}$ , acquisition periodicity  $p^{acq}$  and maximum  
 134 latency  $\lambda$ . The unit is the second (or millisecond, if required).

135 For clarity, we do not present  $\beta, p^{upd}$  in this paper. We then represent  
 136 application requirements on sensors at a given time instant by a set of pa-  
 137 rameterized subscription requests  $S = \{s_1, s_2, \dots, s_n\}$ , where:

$$138 s_i = (d_i, m_i, p_i^{acq}, \lambda_i) \in D \times M \times \mathbb{N}^+ \times \mathbb{N}^+$$

139 **Example 1.** Suppose two applications with the following requirements on 6  
 140 sensors:

141 **Application 1:** Temperature ( $m_T$ ) of sensors  $d_1, d_3, d_5$  with an acquisition

142 periodicity of 2 sec and a latency of 4 sec;  
 143 **Application 2:** Humidity ( $m_H$ ) of sensors  $d_3$ ,  $d_4$ ,  $d_5$ ,  $d_6$  with an acquisi-  
 144 tion periodicity of 5 sec and a latency of 3 sec.  
 145 Based on the previous notation, all application requirements are represented  
 146 by a set of parameterized subscription requests:  
 147  $S = \{(d_1, m_T, 2, 4), (d_3, m_T, 2, 4), (d_5, m_T, 2, 4), (d_3, m_H, 5, 3), (d_4, m_H, 5, 3),$   
 148  $(d_5, m_H, 5, 3), (d_6, m_H, 5, 3)\}$

### 149 3.2. Network Topology

150 To transmit data to the central base station (where results are com-  
 151 puted), we suppose that deployed wireless sensor devices construct a logical  
 152 tree (tree topology) where each sensor has only one neighbor to deliver data  
 153 packets. This tree structure provides a uni-path for each sensor to reach  
 154 the base station (sink device). In fact, our optimization algorithm relies on  
 155 known and unique paths, as it does not consider a choice of network paths  
 156 between two devices. Integrating more complex network topologies, such as  
 157 mesh networks, would be a future work. However, the tree topology may be  
 158 updated each time application requirements change, as it triggers a new op-  
 159 timization. For instance [5] proposes a new technique to organize the sensor  
 160 devices that is likely preferable for outdoor monitoring systems: P-SEP (a  
 161 prolong stable election routing algorithm). The authors consider two-level  
 162 sensor device heterogeneities: advanced and normal devices. They propose  
 163 a clustering mechanism and present new cluster head selecting policy to ex-  
 164 tend the lifetime of the system (lifetime of the system is evaluated in terms  
 165 of FND (First Node Dies)). P-SEP puts forth efficient simulation results and  
 166 network lifetime. Although we do not tackle routing issues, applying our  
 167 dynamic network behavior due to the dynamicity of the application require-  
 168 ments, with the P-SEP mechanism could produce concrete improvements  
 169 on the lifetime of the system. In our case, with our assumptions about the  
 170 network, each sensor device has its own role in the topology based on the  
 171 current applications and its location: **Source** (responsible for the data ac-  
 172 quisition and transmission) and/or **Relay** (responsible for the reception and  
 173 re-transmission), or **Sink** (responsible for only the reception).

174 **Example 2.** A tree topology with 6 multi-modal sensor devices and a base  
 175 station is illustrated in Figure 2. A tree topology has a layered form based on  
 176 the distance to the base station.

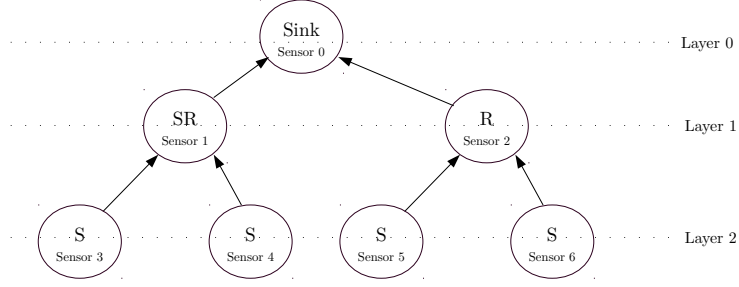


Figure 2: A tree-structured topology and different sensor device roles: Source (S), Relay (R), Source+Relay (SR), Sink.

### 177 3.3. SCO-PATTERN

178 The idea behind the 3SoSM approach is to create a **Schedule Time Pattern**  
 179 for each sensor device. This pattern, described in [6] is a schedule composed of  
 180 sensor actions, here with three types of actions: **acquisition A**, **reception**  
 181 **R**, **transmission T**. Based on their roles, only certain actions are pertinent  
 182 to sensor devices. Sensor devices can configure themselves with a Schedule  
 183 Time Pattern, and then execute periodically the scheduled actions: acquire  
 184 a measure and store it in a memory buffer, transmit all buffered measures  
 185 one hop towards the sink (and empty the buffer), receive measures from a  
 186 lower-layer device and store it in a memory buffer.

187 We call this pattern **sensor configuration oriented pattern** or SCO-  
 188 PATTERN. This pattern consists of timestamped events ( $< timestamp, action >$   
 189 couples) and a length  $\ell$  (or periodicity) of that pattern. These actions are  
 190 enclosed by the length of the pattern: event timestamps are in time interval  
 191  $[0; \ell[$ . A SCO-PATTERN is denoted by:  
 192  $P = (\{(t_i, a_i)\}, \ell)$  with  $\ell \in \mathbb{N}^+$ ,  $t_i \in [0; \ell[$ ,  $a_i \in \{A_m, T, R\}$  where  $A_m$  indicates  
 193 the data acquisition of the physical measure  $m$  such as  $A_T$  for temperature,  
 194  $A_H$  for humidity, etc.

## 195 4. 3SoSM: Energy-Aware Optimization Process

196 To optimize the energy consumption of the devices, since the communica-  
 197 tion costs are the most significant part of the energy budget [7], we want to  
 198 minimize the number of transmission and reception actions of each device:  
 199 less communication means less consumed energy. We first construct “pre-  
 200 liminary” SCO-PATTERNS that have a common length with only acquisition  
 201 actions (Acq) for all the devices. It is then necessary to insert the trans-  
 202 mission (Tx) and reception (Rx) actions in these SCO-PATTERNS to obtain  
 203 a solution, composed of the set of complete SCO-PATTERNS. A solution is



valid if all acquired data traverse the network topology from their source device to the base station before the “latency” associated with that data expires. The optimal solution is a valid solution that minimizes the number of Transmission / Reception actions of each device.

The objective of the optimization process is to transform the application requirements into a global schedule and then to build individual SCO-PATTERNS to configure each device. Application requirements are represented by a set of subscription requests  $S = \{s_1, s_2, s_3, \dots\}$ . Hence the energy-aware optimization process can be expressed as a transformation function that gets a set of subscription requests (all subscription requests from all applications) and a network topology  $\mathcal{N}$  and produces a SCO-PATTERN for each device:  $f(\{s_1, s_2, \dots, s_n\}, \mathcal{N}) = \{P_{d_1}, P_{d_2}, \dots, P_{d_m}\}$ .

#### 4.1. Preliminary SCO-PATTERN

The preliminary SCO-PATTERN is specific for each sensor device. It is periodic, and its length indicates the periodicity. Each sensor device may have different acquisition periodicities from different applications. However, in order to build a global view of the network, a common length is required: it is the lowest common multiple of acquisition periodicities from all subscription requests. Moreover, the length of the schedule should be greater than any latency of the subscription requests to properly handle data expiration time: The final length is the lowest multiple of this initial common length greater than any data expiration time.

**Example 3.** *The common length of the patterns (for the given applications in Example 1) is calculated as:*

$$\text{Initial common length } \ell_{min} = LCM(2, 2, 2, 5, 5, 5, 5) = 10$$

$$\text{Maximum latency } \lambda_{max} = MAX(4, 4, 4, 3, 3, 3, 3) = 4$$

$$\text{Final length } \ell_{schedule} = 10 \text{ sec} > \lambda_{max}$$

Then, based on the periodicity of subscription requests, acquisition actions are defined. Furthermore, each acquisition action is tagged with its maximum latency (from the subscription request).

**Example 4.** *Preliminary SCO-PATTERN for  $d_3$  with subscriptions*

$$\{(d_3, m_T, 2, 4), (d_3, m_H, 5, 3)\} :$$

$$P = (\{(0, \{A_T^4, A_H^3\}), (2, A_T^4), (4, A_T^4), (5, A_H^3), (6, A_T^4), (8, A_T^4)\}, 10)$$

## 237 4.2. *Transmission Constraint and Optimization Goal*

238 To form a global schedule, we adopt a similar approach to Galpin et  
239 al. [8]. A predefined granularity determines the slots in the time space: the  
240 schedule is divided into time-slots and each slot may be filled with sensor  
241 actions. For the following examples, granularity is set to 0.5 *sec* (e.g. 20  
242 slots for 10 seconds).

243 The major constraint of the system is that all the acquired data must  
244 pass through the network topology up to the base station before the expi-  
245 ration of their “latency” in order to fulfill the application requirements. A  
246 transmission action on a sensor device during a single time slot stands for a  
247 transmission of all the data (acquired by itself and received from lower-layer  
248 neighbors) not yet sent, including acquisitions on the same time slot. This  
249 transmission action requires a reception action at the same time slot on the  
250 sensor device that receives the data. Besides there are other constraints due  
251 to the radio protocol such as a device can only receive data from a single  
252 device at a time on the same time slot and a device can not receive and send  
253 on the same time slot.

## 254 4.3. *Searching Optimal Communication Slots*

255 We propose an optimization algorithm to choose the optimal communi-  
256 cation slots from the preliminary SCO-PATTERN. The optimization process  
257 starts by propagating transmission and reception constraints due to the ac-  
258 quisition events in a bottom-up process, and then analyzes the possible re-  
259 ception actions of the base station and tries to find the most energy efficient  
260 communication slots, and finally propagates those choices to lower layers and  
261 continues the analyze in a top-down process.

### 262 4.3.1. *Schedule Table Initialization*

263 We use a schedule table to represent the sensor action slots for Tx/Rx.  
264 Each sensor device has its own schedule and each schedule is composed of  
265 two parts: transmission and reception part. The length of the schedule is  
266 defined by the common length of preliminary SCO-PATTERNS. Each line  
267 of the schedule table corresponds to an acquired measure: its transmission  
268 action for the transmission part and its reception action for the reception  
269 part.

270 The transmission part indicates the possible transmission slots for the  
271 sensor device for each acquired measure. According to the latency informa-  
272 tion  $\lambda$ , every sensor device has the initiative to keep the data (instead of

273 sending it immediately) and send it at an optimal later moment under the  
 274 condition that this acquired data arrives at the base station before the ex-  
 275 piration of its latency. The reception part represents the possible reception  
 276 slots for each data coming from lower-layer neighbors (zero, one or several  
 277 depending on the network topology). Schedule Table for Sensor 3, Sensor 1  
 278 and Sensor 0 (sink device) are illustrated in Figure 3.

279 We propagate bottom-up constraints for data transmission and reception  
 280 based on latencies (from preliminary SCO-PATTERNS) and on the network  
 281 topology.

282 *Transmission Part.* Information from the preliminary SCO-PATTERN is used  
 283 to fill the transmission part. For a given sensor device, each acquired data  
 284 and each received data is represented as a line in the transmission part. A  
 285 line is filled with the residual latency information starting from the time slot  
 286 where the device acquires the data or from the time slot following the time  
 287 slot where the device received the data, and until residual latency reaches 0  
 288 (in fact, 1 time slot before at 1 hop from the sink, 2 time slots at 2 hops,  
 289 etc.).

290 For each device  $d$ ,  $T_d = [e_{i,j}]$  is a 2D matrix with dimensions  $m_d \times n$ .  $m_d$  is  
 291 the number of data to transmit and  $n$  stands for the number of time slots.  $e_{i,j}$   
 292 represents a potential transmission event of data  $i$  at time slot  $j$ :  $e_{i,j} = \lambda_{i,j}$   
 293 is the residual latency,  $e_{i,j} = 0$  means that data  $i$  can not be transmitted at  
 294 that time slot.

295 **Example 5.** *Transmission parts of the schedules of Sensors 3 and 1 are*  
 296 *presented respectively in Figure 3a, 3b. For instance, Sensor 3 measures*  
 297 *temperature and humidity values. For A2#1 (first acquisition for humidity),*  
 298 *the latency is 3sec. Hence, to send that data, Sensor 3 has 5 slots ( $t=0, 0.5, 1,$*   
 299 *1.5, 2 sec) before the latency is expired as it is 2 hops from sink. Transmission*  
 300 *part of the schedule of Sensor 1 covers transmission of its own data and also*  
 301 *received data from Sensor 3 and 4 (see Fig 3b).*

302 *Reception Part.* For a given sensor device, the reception part is based on  
 303 the transmission part of the schedules of lower-layer neighbors. Each line  
 304 represents a data to receive and is linked to one line from the transmission  
 305 part of the source device. Possible reception slots are filled with the data  
 306 residual latency from the corresponding transmission time slot.

307 For each device  $d$ ,  $R_d = [e_{i,j}]$  is a 2D matrix with dimensions  $m_d \times n$  where  
 308  $m_d$  is the number of data to receive and  $n$  stands for the number of time slots.

Acquisition	Measure	Time(sec)																	
		0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5
Acq of S1A1	temperature	A1#1				A1#2				A1#3				A1#4				A1#5	
Acq of S1A2	humidity	A2#1								A2#2									

(a) Schedule of Sensor 3 (Acquisition + Transmission).

Acquisition	Measure	Time(sec)																	
		0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5
Acq of S1A1	temperature	A1#1				A1#2				A1#3				A1#4				A1#5	
Acq of S1A2	humidity	A2#1								A2#2									

(b) Schedule of Sensor 1 (Acquisition + Reception + Transmission).

Reception	From	Time(sec)																	
		0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5
RX of S1A1#1	S1	4	3.5	3	2.5	2	1.5	1	0.5										
RX of S1A1#2	S1																		
RX of S1A1#3	S1																		
RX of S1A1#4	S1																		
RX of S1A1#5	S1	2	1.5	1	0.5														
RX of S1A2#1	S1	3	2.5	2	1.5	1													
RX of S1A2#2	S1																		
RX of S1A2#3	S1	3	2.5	2	1.5	1													
RX of S1A2#4	S1																		

(c) Schedule of Sensor 0 (Sink device) (Reception).

Figure 3: Schedules (highlighted slots are the optimal communication slots based on our process).

309  $e_{i,j}$  represents a potential reception event of data  $i$  at time slot  $j$ :  $e_{i,j} = \lambda_{i,j}$   
 310 is the residual latency,  $e_{i,j} = 0$  means that data  $i$  can not be received at that  
 311 time slot.

312 **Example 6.** For the given example, reception parts of the schedule for Sensor  
 313 1 and Sink are represented respectively in Figure 3b and 3c. For instance,  
 314 reception part of Sensor 1 represents the possible reception slots for each  
 315 acquired data of Sensor 3 and 4 with their residual latency.

#### 316 4.3.2. Optimal Choice of Tx/Rx Slots

317 For each device  $d_i$ , potential transmission and reception slots are now  
 318 identified in their schedule table (matrices  $T_{d_i}$  and  $R_{d_i}$ ). The optimization

process should now choose the most energy-efficient slots, in order to minimize the number of Tx/Rx slots for each device. From the reception part of the schedule, the algorithm tries to group the receptions from the same sensor device. For each time slot, it calculates the total number of possible receptions for each neighbor. The algorithm searches the latest slot in which maximum number of reception can occur. The latest slot indeed creates more opportunities for the lower-layer devices. Otherwise, it would force the lower-layer devices to send the data earlier.

In our example, the algorithm analyzes reception part of Sensor 0 (Figure 3c). It starts by deciding the best communication slot for Sensor 1 and then for Sensor 2. For Sensor 1, the algorithm chooses  $t=7.5$  since the maximum of possible receptions is 6 and the latest occurrence of 6 is at  $t=7.5$ . Slot occupancy is updated: the slot for  $t=7.5$  becomes occupied. Then the algorithm repeats the same decision process for Sensor 2 and chooses  $t=7$ , as the latest possible occurrence of value 4 is at  $t=7$ . Although  $t=7.5$  is the actual latest maximum, this slot is already occupied, so the communication slot for Sensor 2 is set to  $t=7$ . The status of the process after the first iteration is given in Figure 4. The algorithm updates the reception part of the device by defining the unique Rx slot for each concerned data, then it repeats the same process to decide communication slots for remaining data. This iteration continues until all data have a defined Rx slot.

	Time (sec)																	
From	0	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5	6	6.5	7	7.5	8	8.5
S1	3	6	6	6	5	6	4	4	3	4	4	6	5	6	6	6	3	4
S2	1	4	4	4	3	4	2	2	1	2	2	4	3	4	4	4	1	2
slot occupancy	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	X	-	-

Figure 4: Searching for Optimal Rx Slot on Sensor 0  
Slot occupancy: Free -, Occupied X.

Our optimization algorithm is a greedy algorithm [9]. Greedy algorithms are often used in ad hoc mobile networking to efficiently route packets with the fewest number of hops and the shortest delay possible. Here, the best solution for Rx slots is found locally on the sink device. Chosen Rx slots are propagated to the Tx part of lower-layer neighbors in the network topology. Tx/Rx constraints are then locally updated and the process iterates for those devices. This top-down process continues until all Tx/Rx slots are defined for all devices. The finalized reception part of Sensor 0 is illustrated in Figure 3c. The green slots are the optimal slots found by our algorithm. Finalized schedules of Sensor 1 and Sensor 3 are also presented in Figure 3.

#### 350 4.4. Building Complete SCO-PATTERN

351 Once schedules are finalized after searching optimal communication slots  
 352 for each device, the last step is to extract Tx/Rx events to obtain the com-  
 353 plete SCO-PATTERNS.

354 **Example 7.** *Here are the SCO-PATTERNS for the sensor devices of our ex-*  
 355 *ample:*

356  $P_1 = (\{(0, \{A_T\}), (1.5, \{R\}), (2, \{A_T, R\}), (2.5, \{T\}), (4, \{A_T\}), (6, \{A_T\}), (6.5, \{R\}), (7, \{R\}), (7.5, \{T\}),$   
 357  $(8, \{A_T\}), (9, \{R\}), (9.5, \{T\})\}, 10)$   
 358  $P_2 = (\{(0.5, \{R\}), (1, \{R\}), (1.5, \{T\}), (5, \{R\}), (5.5, \{T\}), (6, \{R\}), (6.5, \{R\}), (7, \{T\})\}, 10)$   
 359  $P_3 = (\{(0, \{A_T, A_H\}), (2, \{A_T, T\}), (4, \{A_T\}), (5, \{A_H\}), (6, \{A_T\}), (7, \{T\}), (8, \{A_T\}), (9, \{T\})\}, 10)$   
 360  $P_4 = (\{(0, \{A_H\}), (1.5, \{T\}), (5, \{A_H\}), (6.5, \{T\})\}, 10)$   
 361  $P_5 = (\{(0, \{A_T, A_H\}), (1, \{T\}), (2, \{A_T\}), (4, \{A_T\}), (5, \{A_H, T\}), (6, \{A_T, T\}), (8, \{A_T\})\}, 10)$   
 362  $P_6 = (\{(0, \{A_H\}), (0.5, \{T\}), (5, \{A_H\}), (6.5, \{T\})\}, 10)$

363 Once SCO-PATTERNS are completed, they are sent by the Gateway to  
 364 sensor devices to configure them. Sensor devices then execute the given ac-  
 365 tions periodically, and data flow to the base station where continuous queries  
 366 are computed. If an application changes its requirements, the configuration  
 367 process starts again from the beginning.

## 368 5. Experimental Platform

369 In this section, we present the tools used in this study: **SoCQ** Engine as a  
 370 continuous query engine and **WSNet** simulator for the wireless sensor network  
 371 environment.

### 372 5.1. Continuous Query Engine: SoCQ Engine

373 As a framework for *PEMS*, we use the **SoCQ** (**S**ervice-oriented **C**ontinuous  
 374 **Q**uery) framework [1]. It takes a data-oriented perspective on the pervasive  
 375 environment: it provides a unified view and access to the various and het-  
 376 erogeneous resources available in the environment. Pervasive applications  
 377 can then be created in a declarative way using service-oriented continuous  
 378 queries over such an environment.

379 Within the **SoCQ** framework, *XD-relations* (eXtended Dynamic Relations)  
 380 represent standard relations, that may be updated, or data streams, that  
 381 continuously produce data. The definition of *XD-relations* can also include  
 382 virtual attributes and binding patterns that together enable queries to inter-  
 383 act with distributed services: service discovery, method invocation, stream

384 subscription. Queries may be one-shot queries (like standard SQL queries)  
 385 or continuous queries (with a dynamic result, like a stream). Furthermore,  
 386 invocations of service methods and subscriptions to service streams can be  
 387 parameterized.

388 We illustrate SoCQ in the context of Smart Building. Table 1 shows a  
 389 discovery query, the resulting XD-Relation, then a one-shot query and a  
 390 continuous query over this XD-Relation. The discovery query (**DISCOVER**  
 391 **Services**) searches for sensor services that provide a location, a method to  
 392 get the current temperature, and a continuous stream of temperature mea-  
 393 sures. The resulting XD-Relation **TemperatureServices** has one ServiceID  
 394 attribute (here, the URI of a sensor), a Location, and a virtual attribute for  
 395 Temperature. When executed, the one-shot query (**SELECT ONCE**) selects  
 396 services located in a given room and retrieve the current temperature by  
 397 invoking **getTemperature** method. While executing, the continuous query  
 398 (**SELECT STREAMING**) subscribes to the **temperature** stream of every discov-  
 399 ered service to build a resulting data stream. If new services are discovered  
 400 and/or some services become unavailable, the continuous query automati-  
 401 cally adapts the corresponding stream subscriptions.

Table 1: Example of SoCQ queries for a Smart Building.

CREATE RELATION TemperatureServices ( ServiceID SERVICE, Location STRING, Temperature NUMBER VIRTUAL ) USING ( getTemperature[ServiceId]():(Temperature), temperature[ServiceId]():(Temperature) STREAMING AS DISCOVER SERVICES PROVIDING PROPERTY Location STRING, METHOD getTemperature ( ) : ( NUMBER ), STREAM temperature ( ) : ( NUMBER )       )	SELECT * ONCE FROM TemperatureServices WHERE Location = "501.340" USING getTemperature;	SELECT * STREAMING UPON insertion FROM TemperatureServices USING temperature[1];
---	--	---

## 402 5.2. WSN Simulator: Modified WSN<sub>et</sub>

403 In our platform, we integrated the WSN simulator WSN<sub>et</sub> [10]. WSN<sub>et</sub> is  
 404 a modular event-driven simulator, more precisely a discrete event simulator  
 405 (DES). WSN<sub>et</sub> adopts basic functionality of *DES*: in order to avoid simulating  
 406 every time splice, the time line is split into events and no change is presumed  
 407 to occur in the system between consecutive events; thus the simulation can  
 408 directly jump in time from one event to the next. However, as the PEMS  
 409 works on real-time, **we modified the time scheduler** of the WSN<sub>et</sub> simu-  
 410 lator to avoid time scheduling difference: we introduced a time factor to run  
 411 experiments in real-time or  $n$ -times faster ( $\times 10$ ,  $\times 20 \dots$ ).

### 412 5.3. Gateway: 3SoSM Gateway

413 In a declarative monitoring architecture (see Figure 1), the **Gateway** is a  
414 technical bridge between two environments: it manages interactions and bidi-  
415 rectional communication between the **PEMS Query Engine** and the **WSN**. We  
416 implemented the 3SoSM principles in a **3SoSM Gateway**. It is implemented  
417 in Java, and interacts with the **SoCQ** engine and **WSNet**. 3SoSM Gateway has  
418 two primary modules: the **Service Manager**, that manages **SoCQ** services  
419 representing available sensor devices (real devices or simulated devices); and  
420 the **Subscription Manager**, that continuously analyses application require-  
421 ments to generate new SCO-PATTERNS for sensor devices when required. In a  
422 typical scenario, multiple applications launch continuous queries concerning  
423 sensors to the *PEMS*. The gateway manages required parameterized stream  
424 requests and, according to the 3SoSM optimization process, generates optimal  
425 sensor configurations. With our integrated prototype, we can execute mul-  
426 tiple **SoCQ** queries that dynamically subscribe to streams provided by **WSNet**  
427 sensor devices. The **Gateway** transparently optimizes the subscriptions and  
428 configures the **WSNet** sensor devices with SCO-PATTERNS.

## 429 6. Experiments

### 430 6.1. Experimental Setup

431 The simulations are performed using the **modified WSNet**. We simulate  
432 one part of the topology of our physical platform **SoCQ4Home** deployed in our  
433 LIRIS laboratory: 70 sensor devices are located at specific positions over  
434 a floor of the building (10m × 60m × 4m). The topology is illustrated  
435 in Figure 5. The deployed sensor devices have fixed positions during the  
436 simulation and we consider that they have enough energy until the end of  
437 the simulation. During the simulation process, the initial energy level of  
438 sensor devices is set on purpose to emphasize the lifetime difference between  
439 both cases.

440 We adopt known pervasive environment communication protocol **Zigbee**  
441 **IEEE 802.15.4**, simulated with a **UDG** propagation model (which is a strong  
442 simplification for building environments, as it does not take into account en-  
443 vironmental effects), 35m transmission range, and basic radio module states  
444 for devices (idle, active, sleep, transmission, reception). Calculation of en-  
445 ergy consumption is based on CPU and radio components, adopted from [7]:  
446  $\mathcal{E}_{device} = \mathcal{E}_{CPU} + \mathcal{E}_{radio} = \mathcal{E}_{CPU} + (\mathcal{E}_{sleep} + \mathcal{E}_{TX} + \mathcal{E}_{RX}), with \mathcal{E}_{component/mode} =$



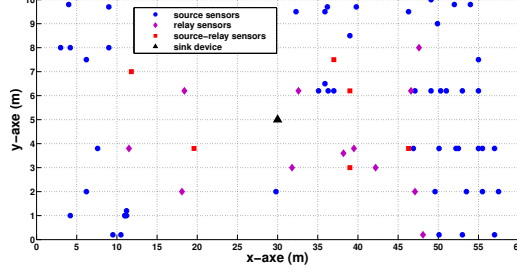


Figure 5: Topology of the network  
(Sensor roles are set based on the Application 1 of the scenario given below).

447  $I_{mode} \cdot V_{drain} \cdot \Delta t_{mode}$ . This model is a well-known energy consumption calcu-  
 448 lation model preferred by many researchers. For instance the paper [11] pro-  
 449 poses a temperature-aware task mapping approach for the mapping of multi-  
 450 application to NoC-based many-core systems by balancing workloads among  
 451 cores. In our study, we adopt a similar approach to calculate the energy con-  
 452 sumption of all devices. We implemented sensor devices for **WSNet** so that  
 453 they can reconfigure themselves when they receive a new **SCO-PATTERN**  
 454 packet, and so that **WSNet** can monitor their simulated energy consumption  
 455 while executing sensor actions.

## 456 6.2. Experimental Scenario

457 To evaluate our approach, we compare 2 types of architecture:  
 458 1) an architecture with basic duty-cycle **WSN** devices, requiring a static con-  
 459 figuration predefined according to scenario requirements;  
 460 2) an architecture with **3SoSM** approach where **WSN** devices can be dynami-  
 461 cally configured with **SCO-PATTERNS** generated by the **3SoSM Gateway**.

462 We design four scenarios: “Temperature of Occupied Rooms”, “Comfort  
 463 Temperature Range”, “Room occupancy based on CO<sub>2</sub> emission” and “Lu-  
 464 minosity of rooms”. Each scenario is performed during one day (1440min).  
 465 Here, we present only one of the performed scenarios due to space limitations.

466 **Scenario “Temperature of Occupied Rooms”:** One of the features  
 467 of a typical smart building is monitoring temperatures and occupancy of the  
 468 environment. Here we suppose that occupancy sensor devices are not always  
 469 active but has their own duty cycle for detecting the motion.

470 In this scenario, there exists two applications:

471 **Application 1:** Monitor the temperature of each room with a data acqui-  
 472 sition every 15 sec and a latency of 60 sec.

473 **Application 2:** Monitor temperature of occupied rooms with a data acquisition every 1 sec and a latency of 4 sec.

475 Here, *Application 1* requests the temperature from all temperature sensor devices with given application requirements. *Application 2* also requests a temperature with a special constraint: it intends to track the temperature only for occupied rooms. For this application requirement, occupancy and temperature sensor devices are needed.

480 For the duty-cycle approach, we adopt a static configuration. In this case, we consider the worst case situation: occupation may be true at any location during the simulation. Based on this constraint, sensor devices are configured with:  $p^{acq} = 1sec, p^{tx} = 4sec$  (periodic transmission)

484 In our approach, we propose a dynamic sensor configuration based on the real-time context. To respond to *Application 1* and *Application 2*, sets of SoCQ queries are implemented. For *Application 2*, from the occupancy stream, we extract the location attribute of the occupied rooms. Then, we list the temperature sensor devices that are located in these rooms and implement a new stream query to subscribe to those temperature services with given application requirements. Since these are continuous queries, once a room is not occupied any more or an unoccupied room become occupied, lists are refreshed and new subscriptions or unsubscriptions are handled in real-time. Then, the **3SoSM Gateway** creates schedules for each sensor device and generates SCO-PATTERNS to configure sensor devices accordingly.

495 Briefly, *Application 1* receives temperature data from relevant devices under every condition. However, *Application 2* has a remarkable condition: at least one room should be occupied so that it starts to receive temperature data. Here, there are two subscriptions to the same set of temperature sensor devices located in occupied rooms with different application requirements. During the simulation (one day), energy consumption (in terms of joules) of every sensor are monitored periodically and logged.

502 Besides, three more scenarios are performed. Scenario 2 consists of tracking temperature of each room and requesting more frequently temperatures for rooms that are out of the current comfort temperature range. Scenario 3 is based on the occupancy information retrieved from CO<sub>2</sub> emission detected in the room and scenario 4 is related to the luminosity of a classroom. For these scenarios, two applications with different application requirements are executed during a day.

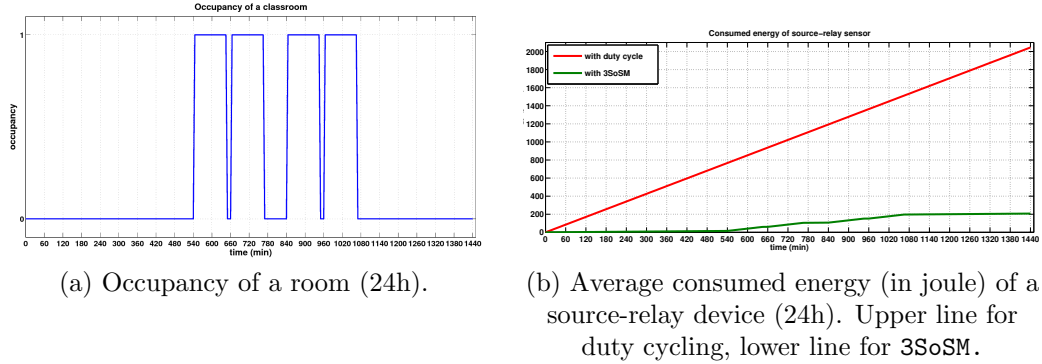


Figure 6: Occupancy information and Energy consumption of the most significant device type: source-relay device with duty cycling and with our approach 3SoSM.

## 7. Results and Discussion

During the experiments, energy consumption of each sensor device is monitored. Figure 6 shows the average energy consumption of source-relay devices. As a remark, the sink device is not considered as a part of this energy consumption calculation. Here, we present the results of the first scenario in detail for one day (1440min). Firstly, the room is not occupied (see Figure 6a), hence the 3SoSM Gateway creates a schedule and configures the relevant sensor devices according to that situation. Once the room is detected as occupied, the 3SoSM Gateway re-creates a schedule and re-configures the relevant sensor devices. In fact, the 3SoSM Gateway creates a new schedule and configures sensor devices to adapt to each contextual change in the environment. Thus, adapting the system to the dynamic context avoids sensor devices from unnecessary data acquisitions and transmissions. Hence, the economy of energy can be achieved with this context-awareness, as shown in Figure 6b.

Evolution of the energy consumption for the WSN can be observed with heat maps as well. Figure 7 represents the heat map of the network based on the energy level of the sensor devices at the end of the simulation ( $t=24h$ ), with duty-cycle method (Figure 7a) and with our 3SoSM approach (Figure 7b), according to the given topology (see Figure 5). With the settings of the scenario “Temperature of Occupied Rooms”, we observe that our approach 3SoSM enhances the lifetime of the sensor devices significantly. For source-relay sensor devices, with given application requirements, 3SoSM extends their lifetime by 8.9 times on average.

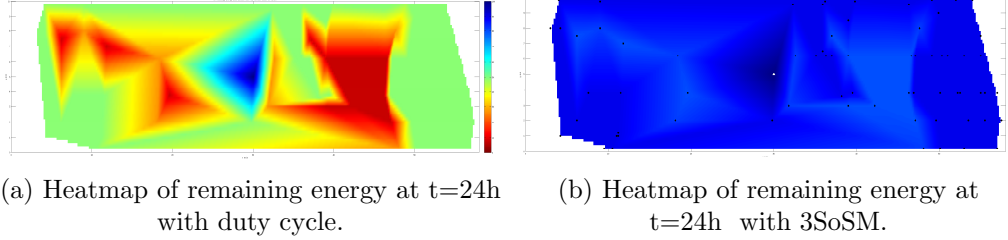


Figure 7: Heatmap of remaining energy with duty-cycle and with 3SoSM.

Table 2: Summary of the Experiments.

Scenarios	Application Parameters				Lifetime (day)	
	Application 1 $p^{acq}$ (sec)	$\lambda$ (sec)	Application 2 $p^{acq}$ (sec)	$\lambda$ (sec)	Duty cycling	3SoSM Approach
Temperature of Occupied Rooms	15	60	1	4	15	146
Comfort Temperature Range	20	60	10	20	9	90
Room Occupancy based on CO2	30	120	15	60	11	95
Luminosity of Rooms	30	100	20	80	10	34

By applying our approach to given application requirements, we optimize the energy consumption and reduce the unnecessary communication cost. Obtained results on energy saving and lifetime extension depend on the application requirements and the context of the application such as temperature and presence. For the current settings, the most unfavorable case is the situation where every room is occupied during the entire simulation time. Even in that case, 3SoSM achieves a concrete energy enhancement since the regular duty-cycle approach does not benefit from latency requirement which provides a tangible energy saving by allowing grouped transmission of multiple acquired data.

Besides, the details of the other performed scenarios are summarized in Table 2 for each scenario, application requirements are given and the average lifetime of source-relay sensor devices are presented. We also obtain a significant energy enhancement and a lifetime extension by avoiding unnecessary communication.

However, balancing the workload of the network over the sensor devices like [11] introduces, advanced clustering mechanism like [5] presents, may further enhance the network lifetime and the obtained results. [5] presents that with a clustering mechanism and selecting cluster headers in an optimal way may extend the lifetime of the network better than any other cluster

553 based networks. With more populated topologies and with massive subscrip-  
 554 tion requests, a discussion about the optimization process appears: in which  
 555 conditions our approach may fail to find the optimal slots on the schedules  
 556 (since a slot is reserved for a single device)? In these cases, granularity should  
 557 be decreased to have enough slots for the sensor devices.

## 558 8. Related Works

559 WSN are related to several different research domains such as pervasive  
 560 environment, and smart building systems. In this paper, we give a brief  
 561 summary of the following research areas that are concerned by our study:  
 562 energy issues in WSN, sensor network query processing (SNQP), sensor-based  
 563 Smart Building Management Systems (SBMS).

564 *Energy Issue in WSN.* Wireless devices bring important constraints such as  
 565 limited battery lifetime. [12] classifies the existing studies on energy con-  
 566 sumption of wireless sensor devices into 3 subgroups: duty-cycle, data-driven  
 567 and mobility-based approaches. Our sensor configuration based subscription  
 568 management approach benefits from scheduled rendezvous of duty-cycling  
 569 and adaptive sampling/transmission of data-driven approaches.

570 *Sensor Network Query Processing.* The main functionality of Sensor Net-  
 571 work Query Processors (SNQP) is to handle continuous queries and sensor  
 572 data streams [13]. [14, 15] propose an adaptive in-network aggregation op-  
 573 erator ADAGA for query processing on sensor devices in order to filter and  
 574 reduce the volume of sensor data. The main functionality of ADAGA is to  
 575 regulate/adjust sensor activities based on energy levels and memory usage  
 576 of sensor devices. As we propose in our own study, ADAGA is also capable of  
 577 processing query parameters but the study does not tackle multi-application  
 578 context and multi-modality of wireless devices as proposed in our own work.

579 We use the Service-oriented Continuous Query (SoCQ) Engine [1] to man-  
 580 age sensor data streams. SoCQ has a strong advantage from its rivals (es-  
 581 pecially SNEE [8]) as it supports multi-application contexts. However, it  
 582 supports neither energy-awareness nor real-time sensor configuration. Our  
 583 approach provides these features and improvement to the SoCQ Engine.

584 Among available technologies for SNQP, [8] is the closest study to our  
 585 approach. This study deal with the conflict between the quality of service and  
 586 sensor acquisition/transmission settings. Authors propose an optimization of  
 587 application requests and the generation of a query execution plan. Generated

588 execution plan provides a global view of the network and a schedule that  
589 shows when to execute a sensor action (sleep, listen, receive/send a packet).  
590 In terms of sensor scheduling and parametrization of a query, this approach  
591 is highly close to ours. Still, query execution plan covers a single query with  
592 a single expectation.

593 *Sensor-based Building Systems.* Some other studies focus on the design and  
594 data management aspects as well. Most of those approaches are proposed  
595 for a specific application and adopt a static configuration for sensor devices.  
596 Sensor-actuator based environment management systems are also studied in  
597 this area. [16] mentions that sensor-actuator interactions has a significant  
598 effect on efficient building monitoring. Moreover, the study points out that  
599 estimation of our daily activities has a crucial role on managing heteroge-  
600 neous sensor devices in the environment and provides energy saving and  
601 longer lifespans. [5] indicates that clustering and energy efficiency have been  
602 considered in wide area of WSN applications and propose a mechanism in  
603 which clusters are dynamically built up by neighbor nodes, to save energy  
604 and prolong the network lifetime.

605 [17, 18] are the closest studies to ours. [17] presents intelligent building  
606 architecture based on a self-adapting intelligent gateway. [18] presents self-  
607 adapting algorithms for context-aware systems. These studies propose dy-  
608 namic system management while processing user preferences. However, these  
609 studies are bounded by predefined building applications and the relation be-  
610 tween application requirements and sensor configuration is not established.  
611 Besides, these approaches do not benefit from the potential reconfiguration  
612 of acquisition and transmission frequencies: sensor configuration stays static  
613 during the system lifetime. Moreover, energy consumption within the WSN is  
614 not considered as a major issue.

615 [19] is also close to our approach. Author implicates event-triggered dy-  
616 namic sensor configuration. The study proposes a ZigBee-based intelligent  
617 self-adjusting sensor platform (ZiSAS) that can autonomously reconfigure  
618 middleware, network topology, sensor density, and sensing rate based on the  
619 environmental situation. However, unlike our proposition, user preferences,  
620 application requirements, query mechanism or data stream processing are  
621 not handled.

622 Overview of most known existing studies on Smart Building applications  
623 is summarized in Table 3. For the categorization of these studies, we ben-  
624 efit from the key functionalities of a Smart Building Management System

Table 3: Overview of most known existing studies on Smart Building applications.

Study	Multi-App	Dynamic User Config.	Context-Awareness	Real-Time Sensor Config.	Energy-Aware Monitoring	Query Engine	Experimentation
<i>Doukas et al.</i> [21]	–	–	✓	–	–	–	Real Testbed
<i>Chen et al.</i> [22]	–	–	✓	–	–	–	Simulation
<i>Byun et al.</i> [19]	–	–	✓	✓	✓	–	Real Testbed
<i>Xiang et al.</i> [23]	–	–	✓	–	–	✓	Real Testbed
<i>Rutishauser et al.</i> [24]	–	–	✓	–	–	✓	Real Testbed
<i>Servigne et al.</i> [25]	–	–	–	–	–	–	Real Testbed
<i>Mamidi et al.</i> [26]	–	–	✓	–	–	–	Real Testbed
<i>Kailas et al.</i> [27]	–	–	✓	–	–	–	Real Testbed
<i>Agarwal et al.</i> [28]	–	–	✓	–	–	–	Real Testbed
<i>Preisel et al.</i> [29]	–	–	–	–	✓	–	Real Testbed
<i>Schor et al.</i> [30]	–	–	✓	–	✓	–	Real Testbed
<i>Li et al.</i> [31]	–	–	✓	–	✓	–	Real testbed
3SoSM	✓	✓	✓	✓	✓	✓	Real Testbed and Simulation

(SBMS) [20]. The last line of the table is allocated for our approach 3SoSM. Our research motivation is that the existing approaches are mostly WSN-level techniques that do not tackle real-time dynamic interactions between application continuous queries and the physical environment, in a multi-application context where sensor devices are multi-modal and requirements are dynamic. Hence, this lack creates a gap between the computing environment and the physical environment, that can be both managed by pervasive applications.

To address this gap and propose a clear solution for multi-application monitoring systems, we design a sustainable multi-application monitoring system architecture for pervasive environments that collects application requirements for sensor data streams and optimizes them into sensor configurations. We propose energy-aware dynamic sensor device re-configuration (as a result of an optimization process) to lower energy consumption while fulfilling real-time application requirements. Compared to existing studies, our approach 3SoSM provides a multi-application mechanism and allows dynamicity for user configurations and network-aware sensor management while optimizing the sensor communication for enhancing energy consumption by real-time sensor configuration.

## 9. Conclusion

In this paper, we focus on one of the major challenges of pervasive monitoring systems like smart buildings: how to optimize/reduce the energy consumption of the monitoring architecture itself while managing sensor data streams. Existing studies do not tackle the energy consumption of the monitoring system and commonly adopt static configurations for sensor devices.

Since application requirements are dynamic with the context, a dynamic sensor configuration can be a suitable option to solve this problem. We introduce a sustainable declarative monitoring architecture where we adopt the PEMS principles to separate application development and optimization of device interactions. We present our approach 3SoSM that is based on a WSN scheduling mechanism for sensor actions. We propose an optimization algorithm to find the optimal communication time slots. As an outcome, an optimized schedule generates SCO-PATTERNS to configure each device. We introduce our implementation 3SoSM Gateway that supports the optimization process for multiple parameterized subscriptions from dynamic application requirements. Our approach is validated by the experiments using the SoCQ Engine and a modified WSN simulator. Moreover, our approach gives opportunities to use real testbed data and simulation data which is not so common in the pervasive environment research domain. Smart building applications are our target application, however, this approach can be applicable to other data-centric pervasive environments as well [2].

## Conflict of Interest

The authors declare that there is no conflict of interests regarding the publication of this paper.

## References

- [1] Y. Gripay, F. Laforest, J.-M. Petit, Socq: A framework for pervasive environments, in: ISPAN'09, IEEE, 2009, pp. 154–159.
- [2] S. Surdu, Y. Gripay, V.-M. Scuturici, J.-M. Petit, P-bench: benchmarking in data-centric pervasive application development, in: Transactions on Large-Scale Data-and Knowledge-Centered Systems XI, Springer, 2013, pp. 51–75.
- [3] S. Servigne, Y. Gripay, O. Pinarer, J. Samuel, A. Ozgovde, J. Jay, Heterogeneous sensor data exploration and sustainable declarative monitoring architecture: Application to smart building, ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci IV-4/W1 (2016) 97–104.
- [4] C. Perera, A. Zaslavsky, M. Compton, P. Christen, D. Georgakopoulos, Semantic-driven configuration of internet of things middleware, in: SKG'13, IEEE, 2013, pp. 66–73.



- 681 [5] P. G. V. Naranjo, M. Shojafar, H. Mostafaei, Z. Pooranian, E. Baccarelli, P-sep:  
682 a prolong stable election routing algorithm for energy-limited heterogeneous fog-  
683 supported wireless sensor networks, *The Journal of Supercomputing* 73 (2) (2017)  
684 733–755.
- 685 [6] O. Pinarer, Y. Gripay, S. Servigne, A. Ozgovde, Energy enhancement of multi-  
686 application monitoring systems for smart buildings, in: *International Conference on*  
687 *Advanced Information Systems Engineering*, Springer, 2016, pp. 131–142.
- 688 [7] O. Pinarer, A. Ozgovde, Improving the energy efficiency of wearable computing units  
689 using on sensor fifo memory, *International Journal of e-Education, e-Business, e-*  
690 *Management and e-Learning* 5 (2) (2015) 105.
- 691 [8] I. Galpin, A. A. Fernandes, N. W. Paton, Qos-aware optimization of sensor network  
692 queries, *The VLDB Journal* 22 (4) (2013) 495–517.
- 693 [9] I. Cardei, M. Cardei, Energy-efficient connected-coverage in wireless sensor networks,  
694 *International Journal of Sensor Networks* 3 (3) (2008) 201–210.
- 695 [10] Wsnet / worldsens simulator, <http://wsnet.gforge.inria.fr/>.
- 696 [11] S. Cao, Z. Salcic, Z. Li, S. Wei, Y. Ding, Temperature-aware multi-application map-  
697 ping on network-on-chip based many-core systems, *Microprocessors and Microsystems*  
698 46 (2016) 149–160.
- 699 [12] G. Anastasi, M. Conti, M. Di Francesco, A. Passarella, Energy conservation in wireless  
700 sensor networks: A survey, *Ad hoc networks* 7 (3) (2009) 537–568.
- 701 [13] F. Jabeen, S. Nawaz, In-network wireless sensor network query processors: State of  
702 the art, challenges and future directions, *Information Fusion* 25 (2015) 1–15.
- 703 [14] A. Brayner, A. L. Coelho, K. Marinho, R. Holanda, W. Castro, On query processing  
704 in wireless sensor networks using classes of quality of queries, *Information Fusion* 15  
705 (2014) 44–55.
- 706 [15] A. Brayner, A. Lopes, D. Meira, R. Vasconcelos, R. Menezes, An adaptive in-network  
707 aggregation operator for query processing in wsn, *Journal of Systems and Software*  
708 81 (3) (2008) 328–342.
- 709 [16] F. Viani, F. Robol, A. Polo, P. Rocca, G. Oliveri, A. Massa, Wireless architectures  
710 for heterogeneous sensing in smart home applications: Concepts and real implemen-  
711 tation, *Proceedings of the IEEE* 101 (11) (2013) 2381–2396.
- 712 [17] J. Byun, S. Park, Development of a self-adapting intelligent system for building energy  
713 saving and context-aware smart services, *IEEE Transactions on Consumer Electronics*  
714 57 (1) (2011) 90–98.

- [18] T. Cioara, I. Anghel, I. Salomie, M. Dinsoreanu, G. Copil, D. Moldovan, A self-adapting algorithm for context aware systems, in: RoEduNet'10, IEEE, 2010, pp. 374–379.
- [19] J. Byun, B. Jeon, J. Noh, Y. Kim, S. Park, An intelligent self-adjusting sensor for smart home services based on zigbee communications, *IEEE Transactions on Consumer Electronics* 58 (3) (2012) 794–802.
- [20] T. A. Nguyen, M. Aiello, Energy intelligent buildings based on user activity: A survey, *Energy and buildings* 56 (2013) 244–257.
- [21] H. Doukas, K. D. Patlitzianas, K. Iatropoulos, J. Psarras, Intelligent building energy management system using rule sets, *Journal of Building and Environment* 42 (10) (2007) 3562–3569.
- [22] H. Chen, P. Chou, S. Duri, H. Lei, J. Reason, The design and implementation of a smart building control system, in: ICEBE'09, Citeseer, 2009.
- [23] S. Xiang, W. Wu, K.-L. Tan, Optimizing multiple data acquisition queries in sparse mobile sensor networks, in: MDM'12, IEEE, 2012, pp. 137–146.
- [24] U. Rutishauser, J. Joller, R. Douglas, Control and learning of ambience by an intelligent building, *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* 35 (1) (2005) 121–132.
- [25] S. Servigne, Y. Gripay, J.-M. Deleuil, C. Nguyen, J. Jay, O. Cavadenti, R. Mebrouk, Data science approach for a cross-disciplinary understanding of urban phenomena: Application to energy efficiency of buildings, *Procedia Engineering* 115 (2015) 45–52.
- [26] S. Mamidi, Y.-H. Chang, R. Maheswaran, Improving building energy efficiency with a network of sensing, learning and prediction agents, in: *Proceedings of the 11th AAMAS, Vol. 1, IFAAMAS'12*, 2012, pp. 45–52.
- [27] A. Kailas, V. Cecchi, A. Mukherjee, A survey of communications and networking technologies for energy management in buildings and home automation, *Journal of Computer Networks and Communications* 2012.
- [28] Y. Agarwal, B. Balaji, R. Gupta, J. Lyles, M. Wei, T. Weng, Occupancy-driven energy management for smart building automation, *Journal of Power* 50 (100) (2010) 150.
- [29] M. Preisel, A. Diaz, W. Wimmer, Energy consumption of smart meters, *Journal of Information and Communication Technologies* (2013) 37.
- [30] L. Schor, P. Sommer, R. Wattenhofer, Towards a zero-configuration wireless sensor network architecture for smart buildings, in: BuildSys'09, ACM, 2009, pp. 31–36.

- 749 [31] C. Li, F. Meggers, M. Li, J. Sundaravaradan, F. Xue, H. Lim, A. Schlueter, Bubble-  
750 sense: wireless sensor network based intelligent building monitoring, Proceedings of  
751 the ICT4S'13 (2013) 159–166.