Context-Aware Machine-to-Machine Communications

Javier Mendonça Costa and Guowang Miao
Department of Communication Systems
KTH Royal Institute of Technology
Email: wir11jme@student.lu.se and guowang@kth.se

***Abstract*—As a key enabler of Internet of things, cellular network based Machine-to-Machine (M2M) communications have been growing rapidly in recent years, being used in a wide range of services such as security, metering, health, remote control, tracking, and so on. A critical issue in M2M communications is the energy efficiency as typically the machine devices are powered by batteries of low capacity and thus, it is the key to optimize their consumption. To achieve higher energy efficiency, this paper proposes the adoption of contexts through a generic context-aware framework for M2M communications. With this framework, machine devices dynamically adapt their settings depending on a series of characteristics such as data reporting mode, QoS features, and network conditions to achieve higher energy efficiency and extend the operating lifetime of M2M networks. Simulation results are provided for four commonly used M2M applications. The results demonstrate considerable energy savings and operating lifetime extension on the network when the proposed context-aware framework is used. Thus, it is shown that contexts play an important role on the energy efficiency of M2M systems.**

Keywords: M2M, Context-Awareness, Energy Efficiency, Internet of Things, MTC.

I. Introduction

The Internet of Things (IoT) is a potential technological revolution that represents the future of computing and communications. In wide-area applications, the success of IoT highly depends on the availability of low-cost Machine-to-Machine (M2M) communications provided by cellular networks. The term M2M communications refers to the communications between machine devices through a telecommunication network without need of human interaction [1]. Standardization and regulator entities such as the 3GPP and ETSI are working on the standardization of M2M communications, but it results in a demanding task due to the vast amount of M2M applications and their diverse traffic characteristics and quality of service (QoS) requirements [2], [3], [4].

Energy efficiency is the key in M2M communications since machine devices are generally powered by batteries. Thus, it is of special interest to optimize the energy consumption and extend the operative lifetime of machine devices. To approach this issue, the new generation of M2M systems requires intelligent algorithms to modify behaviors of machine devices as well as the overall network so they dynamically adapt their settings to improve the energy efficiency. This idea can be achieved by making the machines aware of their states by means of contexts. Contexts have been studied in computing systems to allow devices to adapt automatically to different situations and modify their operative modes for better performance.

Multiple authors have proposed definitions for context and context-awareness [5], referring to context as location, time, environmental information, identities of nearby people or objects including the changes of those objects, and even settings of the application [6]. If a piece of information can be used to characterize a situation of an entity, e.g. person, place, or object then it can be said that the piece of information is a context. As for context awareness, many definitions have been discussed, and names such as adaptive, responsive, and context-sensitive are associated to this term. Schilit and Theimer in [7] say that context-aware applications are those which not only are informed about the context but also react and adapt themselves to it. In [8], context-aware computing is defined as the ability of computing devices to identify, sense, interpret, and react to the environment of a user or the devices themselves. An example of such systems is examined in [9], evidencing how to take advantage of contexts to develop ubiquitous systems.

We would like to find a solution that can improve the energy efficiency of a wide range of M2M applications at a low cost. To achieve this goal, a generic context-aware framework is necessary to bring dynamic adaptations to all M2M devices in response to changes on the contexts observed by the machine devices, which are supported by a set of QoS features and data reporting modes. In this paper, we will define contexts for M2M communications and propose a generic context-aware framework to support energy-efficient M2M communications.

In the following of the paper, we begin in section II with proposing a definition of contexts in M2M communications and discussing the expected input and output of the framework. In section III we continue studying the functioning of the context-aware framework. Later in section IV we present and analyze the results of the system-level simulations. Finally, the conclusions are given in section V.

II. Contexts in M2M Communications

To deliver the generic framework the following questions need to be answered:

- What contexts should be modified? The contexts that can improve energy efficiency and matches the QoS requirement of target applications should be modified.
- What contexts must be analyzed to make adaptation decisions? The contexts that can help to categorize the machine itself and the way it operates should be analyzed.
- How often adaptations are to be performed?

978-1-4799-3088-3/14/$31.00 ©2014 IEEE

- How autonomous must the framework be? The framework should be as autonomous as it can. M2M communications are supposed to be human-free.
- Should the system be open-adaptive or closed-adaptive? With an open-adaptive policy new application behaviors and adaptation plans can be introduced and modified over time. With a closed-adaptive policy there is no support for new application behaviors, the machine can have certain preloaded operative behaviors but no new ones can be added over time [10].

With these questions in mind, we continue defining the input and output parameters of the context-aware framework. The inputs of the framework are contexts related to the machine devices that are analyzed by the machine type communication (MTC) server, which processes the information and responds a set of outputs used by the machine device to alter its operative mode in search of improved energy efficiency.

A. Inputs of the Framework

Considering the concepts of context and context awareness discussed in section I, we define a set of contexts associated to the machine devices and the network, which we have depicted in table I. These contexts are important to determine the state of a machine device within the network, so proper adjustments in its behavior need to be performed.

TABLE I. CONTEXTS, DATA REPORTING MODES AND QOS FEATURES IN M2M APPLICATIONS.

Main Contexts	**Secondary Contexts**
Machine	Machine ID Connectivity Data reporting mode Data source Average packet size Inter-arrival time Location QoS Features Hardware
Network	Cluster ID Data filter

Data Reporting Mode			**Application examples**
Time-driven	Query-driven	Event-driven	
0	0	1	mobile POS
0	1	0	-
0	1	1	-
1	0	0	regular monitoring
1	0	1	home security, telehealth
1	1	0	smart metering
1	1	1	-
QoS Features			
Real Time	Priority	Accuracy	
0	0	0	regular monitoring
0	0	1	smart metering, telehealth
0	1	0	-
0	1	1	-
1	0	0	mobile streaming
1	0	1	mobile POS
1	1	0	-
1	1	1	emergency alerting

To compute the parameters that modify the behaviour of the machine device, the MTC server requires the *Machine ID*, *Data reporting mode*, *QoS Features*, and *Location*.

The *Machine ID* is used to identify the machine device in the network. The MTC server uses this information to store the reports of the machine device in the database.

The *Data reporting mode* lets the MTC server know what kind of application is being dealt with. These modes include time-driven, event-driven, query-driven, or a hybrid combination of these methods [11]. For example the time-driven data reporting mode is used by applications that demand periodic data monitoring. In such method the machine devices activate their sensors and transmitters, capture data and transmit it at a constant periodic time interval. In the event and query-driven methods the machine devices react to certain critical event or query sent by the MTC server.

In table I we have depicted a binary representation of the possible data reporting modes in M2M communications, including examples of well-known applications in the market.

By analyzing the nature of the **Time-driven** applications we conclude that they support modifications of the inter-arrival time, average packet size, packet omissions, and data filtering. Most of the M2M applications fall into this category.

Query-driven applications follow certain instructions from the MTC server transmitting data on request. This type of applications allow packet omissions, as adjacent data reports usually contain redundant information, and thus, extension of the average packet size and inter-arrival time can be done to reduce energy consumption.

Event-driven applications usually transmit data on very specific scenarios. Normally, applications fall into this category when they use priority alarm messages (PAM). The urgent packets in these applications have the highest priority and need to be delivered immediately to the MTC server. Because of the latter they do not support any context modification.

In case of the applications with hybrid data reporting modes, a combination of policies supported by the different modes should be used.

In addition to the differences in the data reporting mode, the QoS features of different M2M applications may also vary dramatically. In table I we have summarized a list of possible combinations of *QoS Features* demanded by different M2M applications. Note that the QoS features are different from conventional understanding of QoS as the focus is on MTC applications.

The **Accuracy** requirement relates to the *data source*, which is considered to be accurate if it is free from errors or omissions. Thus, it is possible to omit packets if the application does not require accuracy, which leads to increased inter-arrival time. On the opposite case, all the packets need to be reported. In some applications the packets can be omitted because of the inherent correlation among the data that has been detected and collected. For example a temperature meter may measure temperature once every hour but the temperature in a whole day varies slightly, and many reporting packets can be omitted depending on the accuracy requirement. It applies similarly in a cluster where many cluster members sensing environmental humidity may report similar readings to the cluster head. The cluster head will compress data and report packets far less than what have been sent by its members.

When accuracy is not required, it is possible to define a transmission factor $\alpha < 1$. As an example of it, we have $\alpha = 0.7$ meaning that 70% of the packets are transmitted, while the remaining 30% omitted. On the other hand, when accuracy is demanded, no packets can be omitted and the transmission factor would be $\alpha = 1$, i.e. all packets are reported. Omitting packet transmissions will lead to modification of both the inter-arrival time and average packet size.

Priority is a feature that applications demand in case they require to transmit PAM, i.e. when alarms are triggered. Due to the urgency it is not possible to support packet omissions, and since PAM is generated by a random event neither inter-arrival time nor average packet size modifications are supported. Nevertheless, such messages could be triggered by thresholds defined by a data filter, which will be discussed in the following section.

The **Real Time** feature is demanded by applications such as surveillance that need constant active connection with the network. This feature is subject to operational real-time constraints, and the application fails if the task is not completed within a deadline. Because of this behavior, it is not possible to support modifications of the inter-arrival time, average packet size or data filtering. However, confined packet omissions can be allowed without affecting the general result.

Finally, using the *Location* of the machine it is possible to detect clusters nearby which can lead to modification of the *Connectivity* and transmitting power of the machine.

B. Outputs of the Framework

We have analyzed the contexts defined in table I recognizing the following as elements that can lead to an extension of the operative lifetime of the machine devices: *Inter-arrival time*, *Average packet size*, *Data filter setting*, *Transmitting power*, *Packet omission*, *Cluster ID* and *Cluster head location*.

By prolonging the *Inter-arrival time*, the time between two consecutive transmissions, it is possible to reduce the amount of transmitting power within a given time interval [12], [13].

By increasing the *Average packet size* of the machine device through merging multiple packets, it is possible to reduce redundant information in both the content and the header of the packets and the frequency the device has to wake up for data transmission. Increasing the average packet size might cause an extension of the inter-arrival time and should be done while assuring the delay requirement.

The *Data filter* is used to filter the information to be transmitted on the network, reducing the amount of data sent by the machine devices and thus, extending their battery life. The following is an example of the data filter, which is a set of thresholds used to trigger data measurements and transmissions.

- Hard Threshold H_T: once the data source value reaches this threshold, the machine device switches on the radio module to transmit the sensed data to the MTC server.
- Soft Threshold S_T: it represents small variations in the data source triggering the machine device to transmit data whenever the data source changes by an amount greater or less than S_T from the last measurement, once the H_T is reached.
- Reporting Threshold T_R: it is the maximum inter-arrival time of a machine device. In case the data source does not meet neither H_T nor S_T for a time equal or greater than T_R, the machine device transmits the last sensed value whether it satisfies the thresholds or not. This policy prevents the machine device to be silent for long periods of time, as well as uncertainty on the network whether the machine is alive or not.
- Time Threshold T_T: it defines how much time will pass until the machine device demands a new context modification request.

Packet Omission can be used to reduce both the amount of packets sent by the machine and the frequency the machine has to wake up. This output is regulated by a transmission factor $0 < \alpha \leq 1$, which represents the percentage of packet transmissions.

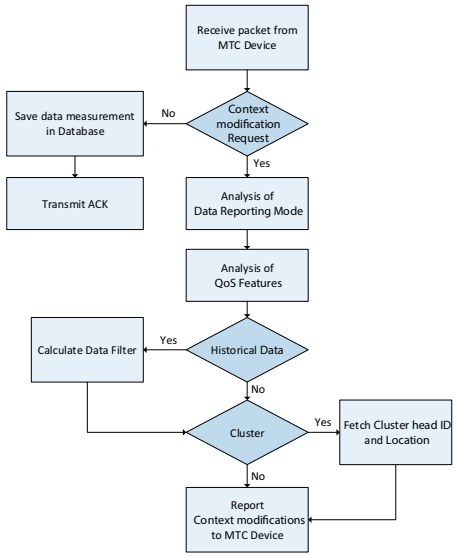
The *Cluster ID* and *Cluster head location* are provided by the MTC server if clustering is used and the cluster groups multiple machine devices together for energy efficient medium access and data transmission.

III. Context-Aware Framework

With the definition of the inputs and outputs of the framework, it is then possible to define the workflow of each element of the system: the machine device, MTC server, and base station (BS).

The machine device first sends a series of contexts to the MTC server. The MTC server analyzes the contexts and responds a combination of suitable context modifications and the process is described in figure 1.

Fig. 1. Flowchart of the MTC Server.

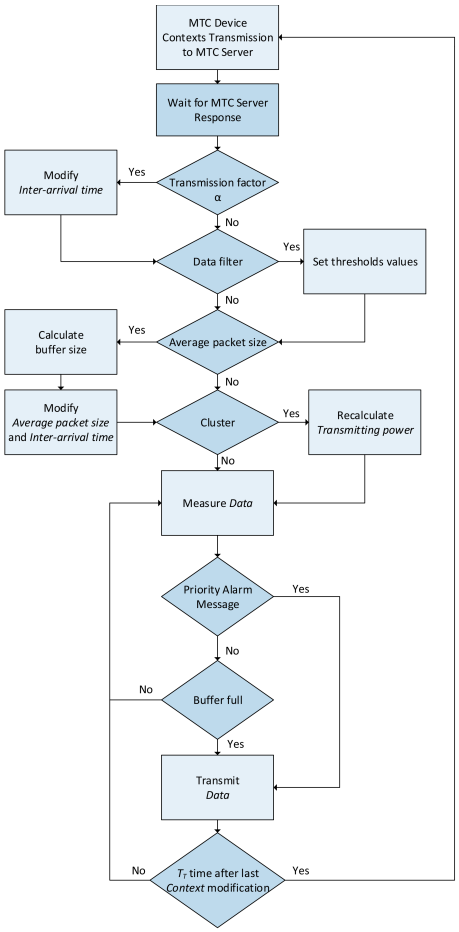


When the MTC server receives a packet, it analyzes if it is a context modification request or just a data report. If the latter then it stores the measurement in the data base and transmits an ACK message to the machine device. If the received packet is a context modification request then it analyzes the *Data reporting mode*, and configures the settings based on the modes that can be supported by the machine device.

Then, it proceeds to analyze the *QoS features* context of the machine. It calculates the transmission factor α for the accuracy demanded by the M2M application. It then sets flags for the rest of the outputs which are supported by the machine according to its *QoS features*. Features such as the *inter-arrival time* and the *average packet size* are supported by both the *Data reporting mode* and *QoS features* and are going to be calculated locally by the machine.

Later, the MTC server uses the *machine ID* to find historical data. If previous data is found, the MTC server computes the H_T, S_T, T_R and T_T as explained previously in subsection II-B. Finally, based on the *location* of the machine device, the MTC server verifies the presence of clusters in the adjacency of the machine which can be formed according to the algorithms studied in [14]. If that is the case the MTC server includes the *cluster head ID* and *cluster head location* in the response so the machine can recalculate its transmitting power. To conclude with the operation, the MTC server replies to the machine device with the appropriate context modifications.

Fig. 2. Flowchart of the Machine Device.



Once the MTC server replies back the context modification request the machine device analyzes the response first by studying if there is any transmission factor α. If so, this factor prolongs the inter-arrival time since a percentage of packets is being omitted. The *inter-arrival time* will be modified as the following expression:

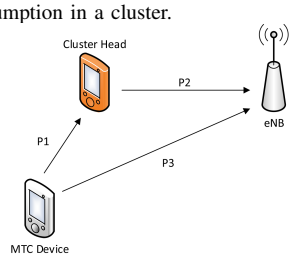
$$I_N = \frac{I}{\alpha} \tag{1}$$

where I_N is the new *inter-arrival time*, I the default *inter-arrival time* of the machine and α the transmission factor computed by the MTC server. We increase the inter-arrival time by omitting and combining packets which decreases the transmitting power while increasing the sleeping power consumed by the machine within a given period of time. Nevertheless, this is a reasonable trade-off since the sleeping power is much lower than the transmitting power.

Then, the machine device verifies if a data filter is to be configured. If so, the machine sets the proper variables, e.g. the H_T, S_T, T_R and T_T. If there is a suitable modification of the *average packet size* the machine itself performs a series of calculations including the available memory or buffer size and the amount of packets P that can be buffered based on the application characteristics in order to find the optimal packet size and reduce the amount of redundant information, e.g. the header overhead. Considering this case, I_N is increased by P times.

The machine device further validates the existence of cluster indication sent by the MTC server. If there is a cluster in the neighborhood the machine recalculates its transmission power lowering it just enough to communicate with the close-distanced cluster head (CH).

Fig. 3. Power consumption in a cluster.



In figure 3, a simple rule of connecting a machine device to a cluster head can be used and it depends on whether the connection reduces energy consumption or not, i.e.

$$(P_1 + P_{c1})t_1 + (P_2 + P_{c2})t_2 < (P_3 + P_{c1})t_3, \tag{2}$$

where P_1 and P_2 are the transmitting powers of the respective links when the data is sent to the BS through the CH. P_3 is the transmitting power when the data is sent directly to the BS. P_{c1} and P_{c2} are the respective circuit powers that are independent of the transmitting states [12], while t_i is the time needed for the reliable transmission of one packet on the corresponding link. Note that the rule in (2) is the same as maximizing the energy efficiency of the link since the connection is used only when

$$\frac{D}{(P_1 + P_{c1})t_1 + (P_2 + P_{c2})t_2} > \frac{D}{(P_3 + P_{c1})t_3}, \tag{3}$$

which is equivalent to

$$\frac{D/(t_1 + t_2)}{(P_1 + P_{c1})\frac{t_1}{t_1+t_2} + (P_2 + P_{c2})\frac{t_2}{t_1+t_2}} > \frac{D/t_3}{P_3 + P_{c1}}, \tag{4}$$

where D is the number of bits in the packet, the left of (4) is the energy efficiency when the machine device is connected to the cluster head and the right the energy efficiency with direct connection to the BS [12]. After the connection is determined, the transmitting power should also be adapted to maximize the link energy efficiency based on [12].

The total energy consumption of the system E_T can be calculated as:

$$E_T = \sum_{i=1}^{N} E_i + E_{CH} \tag{5}$$

where E_{CH} and E_i are the energy consumption of the CH and machines respectively, being N the total number of nodes. The energy consumption of both the machines and cluster head can be formulated as:

$$E = P_{tx}t_{tx} + P_l t_l + P_s t_s \quad (6)$$

where $P_{tx}t_{tx}$, $P_l t_l$ and $P_s t_s$ are the power and the time used by a machine for transmitting, listening and sleeping respectively. The cluster formation, medium access control, and transmitting power should be configured by the BS so that the overall network energy efficiency is maximized [15], [14].

Note that the tasks executed by the CH consume more energy than the ones of a non-CH node. Thus, novel techniques need to be applied where the role of the CH is switched within the members of a cluster so the energy consumption of the system is evenly distributed among all the nodes maximizing the operating lifetime of the network [16]. This issue has been thoroughly investigated in our other work [14], where we have studied the optimal number of clusters in each cell and the optimal reselection of cluster heads to maximize the overall network battery lifetime. We will skip the detailed discussions in this paper because of page limit and the reader is referred to [14] for more information.

IV. Results

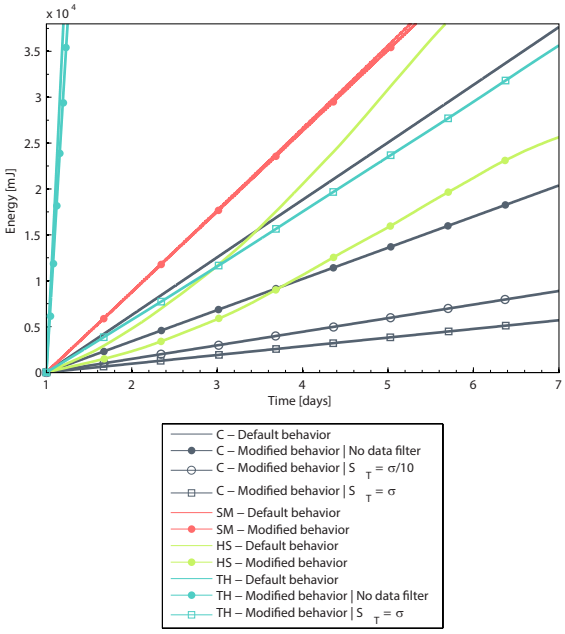
We obtain simulation results of the proposed context-aware framework using Monte Carlo simulations, modeling different random traffic and channel effects for the machine devices in each iteration. The performance metrics include *Average Energy Consumption*, *Energy Savings* and *Operative Lifetime Extension*. We use non-persistent CSMA as the MAC protocol and a traffic model that follows a Poisson distribution. We base our simulations on four major M2M applications with different data reporting modes, including *Climate* (C) (time-driven), *Telehealth* (TH) (time/event-driven), *Home Security* (HS) (time/event-driven) and *Smart Metering* (SM) (time/query-driven) applications. We study the *temperature* parameter for the climate application using real data measurements of a weather station located in Malmö, Sweden captured during the month of June 2013 [17]. The machine devices are stationary and uniformly distributed within the cell. Finally, all the machine devices run the same application and have the same hardware capacities, i.e. are homogeneous. When applicable, packet omissions are performed with a transmission factor $\alpha = 0.5$. PAM are simulated for event-driven reporting mode applications by generating additional packets at random times representing 1% of the total packet transmissions. The data filter threshold values, when applicable, are calculated for a time $T_T = 24$ hours, i.e. new H_T and S_T values are calculated every day based on the measurements of the day before. We calculate the hard threshold H_T as the average of the maximum and minimum values reported on the last T_T, while the S_T are calculated as the standard deviation σ of previous measurements. The T_R is arbitrarily set to 3 times the default inter-arrival time of the respective application. Additional simulation parameters which were chosen according to the LTE uplink budget currently used for MTC [1] are found in table II.

Figure 4 gives the performance of energy consumption for the climate application with periodic data transmissions which characterize regular monitoring applications. As it is shown in figure 5, the proposed context-aware framework obtained

TABLE II. Simulation parameters.

Number of cells	1
Number of BS	1
Number of nodes	23
Number of CH	3
BS height	15 m
BS sensitivity	-136.5 dBm
System margin	26.6 dBm
Bandwidth	61 kHz
Required SNR	-7 dB
Channel capacity	16 kbps
ACK time	1 ms
Max propagation delay	10 μs
Shadow effect	6 dB
Max Transmitting power	23 dBm
Listen power	1 mW
Sleep power	15 μW
Antenna gain	-2 dBi
Buffer size	5 KB
Battery capacity	2700 Joules
Average Packet Size	20 bytes (HS) 128 bytes (TH) 128 bytes (C) 2017 bytes (ST)
Average Inter-Arrival Time	600 sec (HS) 60 sec (TH) 1200 sec (C) 9090 sec (ST)

Fig. 4. Average Energy Consumption - Context-Aware Framework.



energy savings of up to 86% and a lifetime extension factor of up to 6.5 times the default mode, i.e. without context modifications. Even without data filtering, energy savings of 47% are obtained which corresponds to an operative lifetime extension of 1.9 times the case where the framework is not used.

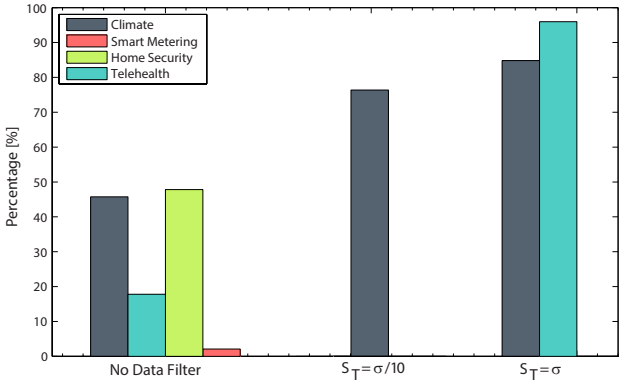
In figure 4 it is also noticeable the influence of S_T on the average energy consumption. The smaller the S_T the higher the energy consumption since the machine devices detect and transmit smaller, more accurate changes in the data source, which in this case is the temperature of the environment. A bigger S_T results in a less accurate outcome, triggering fewer transmissions which results in higher energy savings.

In the case study of the **tele-health application** we analyze the *heart rate* parameter from the data set #12 of vital signs of the University of Queensland [18]. As it can be seen in figure 5 we have accomplished positive results achieving energy savings of 19% and a dramatic 96% without and with the data filter respectively, corresponding to a lifetime extension factor of up to 32 times the default value.

The reason for obtaining lower energy savings in comparison to the climate application when considering the modified behavior without data filter is the *QoS features*; the tele-health application requires accuracy while the climate application does not. Thus, the transmission factor α is set to 1 so no packets are omitted, even when they are redundant.

In addition, the dramatic energy savings obtained by the framework when data filtering is performed has to do with S_T. The data source does not vary too much, i.e. the patient show stable heart rate and few measurements are actually being triggered by the threshold values.

Fig. 5. Energy Savings - Context-Aware Framework.



Finally, for the **smart metering** and **home security applications** we have achieved energy savings of 2.5% and 47% respectively, as seen in 5. No data filter was applied since no data sets were found for these applications. When analyzing in detail the smart metering application, we understand how the buffer size of the machine has a significant impact on the overall energy efficiency. The buffer size used in the simulations was set to 5000 bytes, i.e. about 2.5 packets were combined by the machine in each transmission which means that no significant header reductions were performed. So neither meaningful energy savings nor operative lifetime extension were achieved in this case.

At the same time the home security application uses a hybrid data reporting mode including time-driven and event-driven at the same time. Energy savings of 49% were achieved which corresponds to an operative lifetime extension of 1.75 times that matches the result obtained by the time-driven climate application when no data filter is used.

V. Conclusions

The market of M2M communications in cellular networks is expanding rapidly making the IoT a reality. Work is being done to standardize the way M2M communications operate but there are still many aspects such as the energy efficiency issue that need to be addressed and improved in this field. The proposed framework adopts concepts of context and context-awareness to M2M communications and suggests several contexts that go beyond location to exploit different features of the machine devices. The framework significantly improves the energy efficiency of M2M networks and prolongs the operative lifetime of the network. The system-level simulations performed with major M2M applications show that the usage of contexts in M2M communications is indeed beneficial in improving the energy efficiency of a M2M system with minor performance tradeoffs.

References

[1] R. Ratasuk, J. Tan, and A. Ghosh, "Coverage and capacity analysis for machine type communications in LTE," in *Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th*, pp. 1–5, 2012.

[2] 3GPP TS 22.368 V12.1.0, "Service requirements for Machine-Type Communications (MTC)," tech. rep., December 2012.

[3] R. Liu, W. Wu, H. Zhu, and D. Yang, "M2M-oriented QoS categorization in cellular network," in *Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on*, pp. 1–5, 2011.

[4] P. Zhang, G. Miao, and J. Costa, "M2M communications in lte systems," 2014. submitted for journal publication.

[5] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," 1999.

[6] T. Rodden, K. Chervest, N. Davies, and A. Dix, "Exploiting context in hci design for mobile systems," 1998.

[7] B. N. Schilit and M. M. Theimer, "Disseminating active map information to mobile hosts," vol. 8, no. 5, pp. 22–32, 1994.

[8] N. S. Ryan, J. Pascoe, and D. R. Morse, "Enhanced reality fieldwork: the context-aware archaeological assistant," in *Computer Applications in Archaeology 1997* (V. Gaffney, M. van Leusen, and S. Exxon, eds.), British Archaeological Reports, (Oxford), Tempus Reparatum, October 1998.

[9] J. Pascoe, "Adding generic contextual capabilities to wearable computers," in *Wearable Computers, 1998. Digest of Papers. Second International Symposium on*, pp. 92–99, 1998.

[10] P. Oreizy, M. M. Gorlick, R. N. Taylor, D. Heimhigner, G. Johnson, N. Medvidovic, A. Quilici, D. S. Rosenblum, and A. L. Wolf, "An architecture-based approach to self-adaptive software," *IEEE Intelligent Systems and their Applications*, vol. 14, no. 3, pp. 54–62, 1999.

[11] A. E. K. Jamal N. Al-karaki, "Routing techniques in wireless sensor networks: A survey," 2004.

[12] G. Miao, N. Himayat, and G. Li, "Energy-efficient link adaptation in frequency-selective channels," *Communications, IEEE Transactions on*, vol. 58, pp. 545–554, February 2010.

[13] G. Miao, N. Himayat, Y. Li, and D. Bormann, "Energy-efficient design in wireless OFDMA," in *Proc. IEEE ICC 2008*, pp. 3307–3312, May 2008.

[14] P. Zhang and G. Miao, "Energy-efficient clustering design for M2M communications," 2014. submitted to Proc. IEEE Globecom.

[15] G. Miao, X. Chen, A. Azari, and L. Lu, "Low-power MAC design for M2M communications," 2014. submitted to Proc. IEEE Globecom.

[16] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, 2000.

[17] NOAA's National Climatic Data Center, "NNDC Climate Data Online," June 2013.

[18] The University of Queensland, "The university of queensland vital signs dataset," June 2013.