Context-Aware Machine to Machine Communications in Cellular Networks

JAVIER MENDONÇA COSTA

Master's Degree Project
Stockholm, Sweden October 2013

TRITA-ICT-EX-2013:249

Abstract

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 that needs to be considered in M2M communications is the energy efficiency, typically the machines are powered by batteries of low capacity and it is important to optimize the way the power is consumed. In search of better M2M systems, we propose a context-aware framework for M2M communications so the machine type communication (MTC) devices dynamically adapt their settings depending on a series of characteristics such as data reporting mode and quality of service (QoS) features so higher energy efficient is achieved, extending the operating lifetime of the M2M network. Simulations were performed with four commonly used M2M applications: home security, telehealth, climate and smart metering, achieving considerable energy savings and operating lifetime extension on the network. Thus, it is shown that contexts play an important role on the energy efficiency of a M2M system.

Keywords: M2M, Context-Awareness, Energy Efficiency.

Acknowledgements

I want to thank all the people who supported me not only during the thesis project but the whole master studies, which is something very significant for me. I am very grateful to my family and everybody from Lund University and KTH.

Additionally, I want to thank Dr. Guowang Miao for supervising me in this work and for giving me the opportunity to take this master degree project at KTH, which was indeed very educational and fulfilling.

Contents

List of Figures

List of Tables

List of Acronyms

M2M	Machine-to-Machine
H2H	Human-to-Human
MTC	Machine Type Communication
LTE	Long Term Evolution
ETSI	European Telecommunications Standards Institute
3GPP	Third Generation Partnership Project
QoS	Quality of Service
PDA	Personal Digital Assistant
IoT	Internet of Things
BS	Base Station
eNB	Evolved Node B
CH	Cluster Head
MTCD	Machine Type Communication Device
CSMA	Carrier Sense Multiple Access
PDF	Probability Density Function
PAM	Priority Alarm Message
C	Climate
TH	Telehealth
HS	Home Security
SM	Smart Metering

Chapter 1

Introduction

The term Machine-to-Machine (M2M) refers to the communications between machines through a telecommunication network without need of human interaction [2, 3]. Its market has been growing constantly and it is believed that cellular networks will play a important role in the strong deployment of M2M services, some sources suggest that there will be around 50 billion M2M devices [4, 5] connected by 2020. 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 big amount of applications [6]. M2M communications behave quite different from Human-to-Human (H2H) communications: they show contrasting traffic patterns, different quality of service (QoS) requirements and diverse sets of features [7, 3, 6].

Energy efficiency is the key in M2M communications, since machine type communication (MTC) devices are typically powered by batteries the main goal is to optimize the energy used for transmission. In search of better M2M systems, intelligent algorithms are needed to modify the behavior of the machines so they adapt their settings to improve the energy efficiency. One way to do this is by adopting contexts effectively, a topic which have been studied in computing systems to allow devices to adapt automatically to different situations, modifying their operating modes. Using contexts in M2M communications would allow the machines to achieve higher energy efficiency without any human intervention.

This document presents a possible implementation of context-awareness in of M2M communications.

1.1 Background

Different studies have been made attempting to define different aspects of M2M communications.

In [7], Shafiq *et al.* studied the traffic generated by M2M devices by comparing it with traditional H2H traffic, with the goal of determining if M2M traffic imposes new requirements on cellular networks. Measurements from a cellular network in the United States were taken and analyzed from different perspectives, including network performance, application usage, temporal dynamics and mobility. The results show that M2M traffic differs in many ways from H2H traffic - for instance, M2M traffic shows a much larger ratio of uplink to downlink traffic volume and much shorter periods of activity. It also demonstrates strong daily diurnal pattern, which means that M2M traffic volume peaks are connected to people's working hours while H2H traffic volume peaks coincide with human waking hours.

In [3] it is analyzed the QoS categorization of M2M services based on typical applications in the market. It is evaluated the QoS classes for H2H communications in cellular networks, and it is concluded that they don't fit M2M services since they are oriented to voice and multimedia - which is not common in M2M communications. Finally it is proposed a new set of QoS categories which covers both M2M and H2H services in cellular networks.

Ratasuk *et al.* in [2] made a coverage and capacity analysis for M2M communications in LTE networks. In the studies, authors considered 5 typical M2M applications that can be found in cellular networks nowadays: surveillance, home security, telehealth, smart metering and traffic sensor. With system-level simulations the results show that LTE is capable for MTC by offering vast coverage and large capacity.

To identify potential issues in the air interface, academic work relevant to random access procedures and radio resource allocation and scheduling algorithms in M2M communications has been done in [8, 9, 10, 11, 12, 13]. The analyses showed once again that the characteristics of M2M communications differ in great magnitude from those of H2H communications. In search of the standardization of M2M communications in 3GPP different solutions are suggested in those documents, including grouping-based radio resource management and scheduling algorithms which consider channel conditions and delay constraints of the devices.

There are several studies related to Context-Awareness in computing systems, which can be extended to M2M communications. To make a clear picture of what context and context-awareness is about, Dey and Abowd in [14] recognized the importance of exploiting contexts for interactive applications, surveyed the work of several authors associated to context-aware computing, provided definitions and categories of context and context-awareness and finally, recommended how to use these features to develop context-aware applications.

In cooperation with other authors, Schmidt have studied the relevance of context for years. In [15] he introduced a working model for utilizing contexts in mobile devices such as PDAs and mobile phones to take advantages of the physical environment surrounding the device. In [16] it is described an architecture that supports dynamic context-aware systems which enables context fusion and reasoning. Other studies [17, 18, 19, 20] consider the usage of context-awareness in different scenarios, e.g. health, evidencing how to take advantage of contexts to develop ubiquitous systems.

By following the fundamental definitions of context and context-awareness described in [14], we propose an implementation of context-awareness in M2M communications in cellular networks.

1.2 Problem description

M2M communications have been studied and developed in recent years but it is far from being flawless. Nowadays in the market different providers offer a variety of solutions for various applications employing M2M technology, using their own and unique protocols. Such diversity represents a problem of compatibility that is being solved by the 3GPP and ETSI, but still many tasks have to be done.

One important area of research is the extension of the operating life of the machines, for which it is important to employ proper algorithms to lower the power consumption. As we described in the previous section, important work has been done to develop context-awareness in computing systems, but there are no standards that relate contexts and M2M communications.

This thesis studies contexts and proposes a framework to support context-awareness in M2M communications in cellular networks. Implementing contexts in this frame will improve current systems, allowing to extend the operating life of the machines by modifying their settings such as traffic properties and thus, reduce their energy consumption.

1.3 Goals

This thesis focusing on context-aware M2M communications has the following goals:

- Research on existing context and context-aware concepts.
- Define contexts for M2M communications.

- Exploit the usage of contexts in M2M communications by proposing energy-saving algorithms.
- Design a Context-Aware Framework for M2M communications.

1.4 Methods

In this thesis work we use different scientific methods to conduct the research:

- Literature review: diverse studies in the field of context, context-awareness and M2M communications are studied and accessed through IEEE Xplore, a digital library with thousands of publications on the topic of research.
- Exploratory research: context and M2M communications are topics that have not been clearly defined by the time of this research. The hypothesis of this project is that it is possible to save energy in M2M communications by using contexts, which can modify the behavior of the machines and thus reduce their power consumption, extending the operative lifetime of the network. The hypothesis is to be proofed by theoretical analyses and experimental results.
- Quantitative method: Numeric results are to be analyzed in the experiments, which directly indicates the efficiency of the proposed context-aware framework.
- Experiment: the experiments are to be performed using Matlab, a professional simulation platform used in many fields such as engineering, economics, etc. Its ability to perform matrix computations is an advantage which makes it suitable for this project.

1.5 Scope and Delimitations

This thesis contributes with an implementation of contexts for the design of context-aware M2M communications in cellular networks. The results are based on system-level simulations.

The following limitations are considered in the simulations:

- It is only considered M2M applications with no or low mobility, which is defined in [21] as applications running on devices which are fixed, do not move constantly or may move within a small area, e.g. smart metering, home security, telehealth.
- It is only considered M2M applications with no or low real-time demands.

- No H2H traffic, i.e. only M2M traffic is taken into account, no traffic from user equipments is considered.

For purposes of presentation, the results will be shown with audiovisual material.

1.6 Thesis Structure

The current chapter provides an introduction to the background, problem, goals and methods of this project. We proceed presenting literature studies in Chapter 2, where concepts of M2M communications, context and context-awareness are presented.

In Chapter 3 we propose contexts for M2M communications, as well as different energy-saving algorithms. We continue proposing a context-aware framework for M2M communications which is capable of handling a wide range of M2M applications, analyzing the work flow of each element of the network: MTC device, MTC server and eNB, as well as how the proposed contexts are modified to achieve higher energy savings and extend the operative lifetime of the M2M system.

Then in Chapter 4 we describe the system-level simulations, presenting different simulated scenarios and conditions considered in each experiment. The results are then presented, analyzed and discussed, obtaining important conclusions for this project.

Finally, in Chapter 5 we summarize the project, describing the findings and conclusions obtained with the simulations. As future work, we recommend potential improvements and fields of study for this project.

Chapter 2

Theoretic Background

This chapter discusses concepts and definitions for the interpretation and understanding of this thesis, mainly covering machine-to-machine, context and context-awareness topics.

2.1 Machine-to-Machine Communications

The term M2M communications describe automated data communications between machines and finally between machines and people over a communication network. M2M is compounded by MTC devices and MTC servers. The function of the devices is to gather data and report it back to the server.

2.1.1 General Operation of M2M Applications

Although every deployment might have unique characteristics based on the application, there are four essential operations that are performed in all M2M applications [22, 23]:

- Collection of data: MTC devices use sensors to capture events (i.e. location, temperature, blood pressure, inventory level etc.) and transform the data from analog to digital to be sent over a communications network with IP packets.
- Transmission of data: the data is sent over the network to a remote location, which can be either other device in the network such as a cluster head (CH) or a MTC server. Although this operation can be done by means of wired or wireless networks, typically M2M systems transmit the measured data over a cellular network.
- Evaluation of the data: once the remote location receives the data, it analyzes it and translates it into meaningful information (i.e. fire detected, inventory needs to be restocked, etc.).

- Response to the transmitted information: after receiving and evaluating the data, the remote location might send a reply back to the MTC device.

2.1.2 M2M Application Features and Service Requirements

As it can be seen in [7], M2M applications have different characteristics in comparison to H2H communications which are oriented to multimedia services (e.g. downloading, streaming). A series of features have been identified [6, 4] for M2M communications:

- Low mobility: MTC devices do not move, move rarely or may move only within a small region.
- Small data transmissions: traffic usually consists of small volumes of data.
- Infrequent data transmissions: sporadic traffic allocates network resources only when needed.
- Large number of devices: M2M devices exceed human users by large numbers.
- Time controlled: machines can transmit or receive data only during specific time intervals. Signaling outside of these specific time intervals can be minimized.
- Mobile originated only: the network should provide mechanisms to reduce signaling of MTC devices.
- Packet switched only: the network should provide triggering capabilities with or without a Mobile Station International Subscriber Directory Number (MSISDN).
- Time tolerance: MTC applications can delay data transmissions except specific cases such as priority alarm messages (PAM).
- Priority alarm message (PAM): support of priority alarm messages in case of urgent situations such as theft, vandalism or other special event related to the application.
- Secure connection: secure connection between the MTC devices and the MTC server.
- MTC monitoring: intended for MTC device related events such as loss of connectivity or change of location.
- Group based MTC features: possibility to arrange MTC devices into groups. Different group based features can be applied such as policing, QoS, addressing and charging.

2.1.3 M2M Application Issues

Different challenges can be identified based on the features and service requirements of M2M communications, as for example [24, 25]:

- Energy efficiency: devices are constrained in terms of power, low-power consumption solutions are critical.
- Autonomous operation: MTC devices should be able to maintain and reconfigure themselves without human intervention.
- Diverse traffic characteristics: there are many application with different traffic properties (i.e. inter-arrival rate, delay tolerance, packet size, etc.) which makes it difficult to create a solution to fit all the M2M application requirements.

2.1.4 Traffic Characteristics of M2M Communications

Traffic characteristics in M2M communications vary from application to application, they present different traffic patterns [7] and QoS demands [3], for which it is challenging to say how the M2M traffic is. We can say that the traffic follows a poisson distribution [26, 7, 13], thus the inter-arrival rate is exponentially distributed.

Let I be a random variable that expresses the inter-arrival time between two consecutive packets; the probability density function (PDF) of the exponentially distributed inter-arrival rate is:

$$f_I(t) = \lambda e^{-\lambda t} \tag{2.1}$$

where λ is the average packet-arrival rate in $packets/s$. Nevertheless, to have a more realistic description of the traffic, additional parameters are used to specify the source [1, 27]:

- λ: average data rate.
- σ: maximum burst rate
- α: average packet length in bits.

Given the average inter-arrival time I, it is possible to formulate the average data rate λ and the maximum burst rate σ as [1, 27]:

$$\lambda = \frac{1}{I} \tag{2.2}$$

$$\sigma = \frac{1}{I - \theta} \tag{2.3}$$

where θ is a constant value between 1 and $I-1$.

Consequently, the inter-arrival time distribution function 2.1 is modified as a shifted exponential function as it can be seen in figure 2.1 and can be formulated as [1, 27]:

$$f_I(t) = be^{-b(t-a)}, \text{for } t \geq a,\ a > 0 \tag{2.4}$$

where a is a parameter that stands for the minimum time between consecutive packets and b determines the decay of the exponential function over time. Both the position parameter a and the shape parameter b can be calculated as [1, 27]:

$$a = \frac{\alpha}{\sigma} \tag{2.5}$$

$$b = \frac{1}{\alpha} \times \frac{\sigma\lambda}{\sigma - \lambda} \tag{2.6}$$

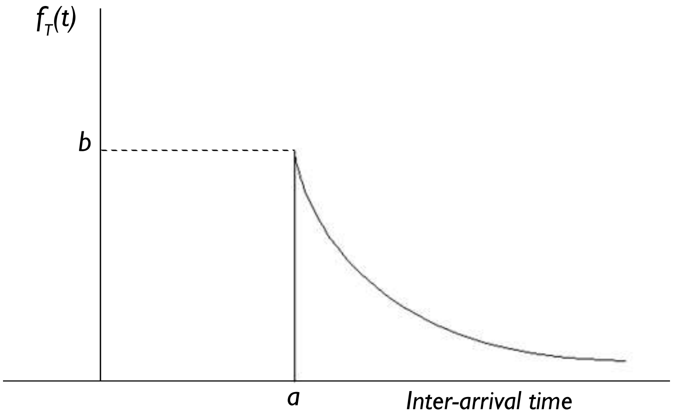

Figure 2.1: Biased exponential distribution with the parameters a and b [1].

2.2 Context and Context-Awareness

There are several definitions of context and context-aware systems [14]. Schilit and Theimer in [28] introduced these terms, where they refer to context as location, identities of nearby people and objects, including the changes of those objects. Other authors state similar definitions of context [29, 30] as location, time, environmental information such as temperature or humidity. Roden *et al.* in [31] define it as the setting of the application.

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. It is important to mention that context can be either implicit or explicit information inputed by the user [14].

As for context, many definitions of context-awareness have been defined, and different names such as adaptive, responsive and context-sensitive among others are associated to this term. Schilit and Theimer in [28] say that context-aware applications are those which not only are informed about the context but also adapt themselves to it. In [17, 30] 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.

2.2.1 Types of Contexts

There are different kinds of contexts defined by different authors, to help application developers exploit as much information as possible. In [30], Ryan *et al.* propose location, environment, identity and time as types of contexts. Schilit *et al.* in [32] emphasizes the important types of contexts as where you are, who you are with and what resources are nearby. In [14] Dey and Abowd name location, identity, activity and time as primary types of contexts. As an example of the latter, if the identity of a smart utility meter is given, then it is possible to get information related to it, such as address or the name of the person who lives where this meter operates. In that case, the identity would be the primary context and the address or name of the person would be a secondary context.

2.2.2 Context-Aware Application Features and Capabilities

Two major taxonomies have been discussed by Schilit *et al.* [28] and Pascoe [17] to categorize the features of context-aware applications. The following is a list of three major attributes that context-aware applications may support [14]:

- Presentation: display relevant information or service to a user.
- Automatic execution: perform certain task automatically based on a context.
- Tagging: label information related to a context for later retrieval.

When considering the most common type of M2M application, i.e. regular monitoring, "Automatic execution" seems to be the most common attribute, while applications such as surveillance are more likely to present relevant information to the user.

Chapter 3

Context in M2M Communications

This chapter proposes contexts for M2M communications, as well as the context-aware procedures and algorithms to be performed by the different elements of the system to improve the energy efficiency.

3.1 Context and Context-Awareness in M2M Communications

We could see in chapter 2 that there are several definitions of context, where all of them see context as environment or situation either of the user or the application. Thus, in M2M communications it can be said that context is any information that can be used to characterize the state of a MTC device.

By analyzing what kind of information can be used by the MTC devices in the design of energy-efficient M2M systems, we define table 3.1 to characterize a set of contexts.

We define a two-tier table structured with two main contexts, one related to the machines and the other one related to the network.

3.1.1 Machine-related Contexts

- Machine ID: needed to establish communication with the network.
- Connectivity: defines if the MTC device is connecting directly to the eNB or to other device. From here it can be inferred if the MTC device is acting as a node or CH in the network.
- Data reporting mode: based on the data reporting method a M2M application can be categorized as time-driven, event-driven, query-

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

Table 3.1: Contexts in M2M communications.

driven, or a hybrid combination of these methods [33]. The time-driven data reporting method is used by applications that demand periodic data monitoring. In such method the MTC devices activate their sensors and transmitters, capture the data and transmit it at a constant periodic time interval. In the event and query-driven methods the MTC devices react to certain critical event or query sent by the MTC server.

- Data source: it is the parameter of interest that is measured by the sensors of the MTC device.
- Average packet size: it is the average length of the data packet created by the machine. The size of it depends on the application.
- Inter-arrival time: defines the time between two consecutive transmitted packets.
- Location: the geographic position of the MTC device as well as date and time.
- QoS Features: describes the demand of the given application of real time, accuracy or priority in the communication.
- Hardware: elements of the MTC device itself that can be used to characterize the machine, such as memory or transmitting power.

3.1.2 Network-related Contexts

- Cluster ID: used to enable group-based communication.
- Data filter: it corresponds to a set of thresholds computed by the MTC server that acts as a filter to decrease the MTC device transmissions so only specific data of interest is reported.

3.2 Context-Aware Algorithms

Next, based on the contexts defined in the table 3.1, we define a series of algorithms to improve the energy efficiency of the M2M communications.

3.2.1 Algorithm 1: Recalculation of Transmitting Power

Based on the context *Connectivity*, the nodes can determine whether there is a cluster or not. In case of existence of cluster, the nodes communicate with the CH instead of the base station. Then the nodes modify the context *Hardware* by recalculating their transmitting power, since the communication range is shorter the power needed to communicate to the CH is lower. If a node is acting as CH then no transmitting power can be saved, since it is the one that communicates to the base station and is responsible of retransmitting the information of the nodes within the cluster.

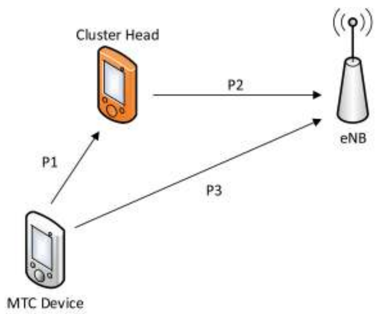

Figure 3.1: Power consumption in a cluster.

In figure 3.1 we see that clustering will only improve the energy efficiency of the network if the following inequality is met:

$$P_1 + P_2 < P_3 \tag{3.1}$$

If it is not met, i.e. the power used by the MTCD to communicate to the CH and later to the eNB ($P_1 + P_2$) is higher than the one used by the MTCD towards the eNB (P_3), then clustering would represent a waste of energy and thus it would be preferable to keep the link MTCD - eNB rather than MTCD - CH - eNB. The total energy consumption of the system E_T can be calculated as:

$$E_T = \sum_{i=1}^{N} E_i + E_{CH} \tag{3.2}$$

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 CH can be formulated as:

$$E = P_{tx}t_{tx} + P_l t_l + P_s t_s \tag{3.3}$$

where $P_{tx}t_{tx}$, $P_l t_l$ and $P_s t_s$ represent the energy consumed by the machine for transmitting, listening and sleeping respectively. Since we assume CSMA as MAC protocol in our design, the machine consumes P_l power during a time t_l to listen to the channel for avoiding collisions, then if the medium is available it consumes P_{tx} power for a period of time t_{tx} to transmit the packets and finally P_s power for t_s time to sleep, switching off the radio module until the next packet transmission.

The transmitting power in mW depends on the channel conditions, which is calculated as the following [10]:

$$P_{tx} = S + L_M + PL + L_S - G_{tx} \tag{3.4}$$

where S is the sensitivity of the receiver in dBm, L_M the system margin in dBm, PL the path loss between transmitter and receiver in dB, L_S the shadow effect and G_{tx} the gain of the transmitter. When the machine communicates directly to the eNB, the pathloss PL can be calculated as [10]:

$$PL = 128.1 + 37.6 \log_{10}(R) \tag{3.5}$$

where R is the distance in m between the transmitter and receiver. On the other hand, when the machine communicates to the CH, the PL is formulated as [10]:

$$PL = 38.5 + 20 \log_{10}(R) \tag{3.6}$$

The listening and sleeping power are fixed parameters which are shown in table 4.1.

3.2.2 Algorithm 2: Adjustment of Inter-Arrival time

There are event-driven applications that have the priority alarm feature and they report data periodically at the same time. Such applications (i.e. home security) have a hybrid data reporting method, in which a priority alarm message is transmitted in case a specific event is triggered (e.g. theft or intruder detection) and in addition, periodical transmissions are executed when they are not really needed. For such applications it is a waste of energy to transmit data during the day when there is nothing critical to report. Based on the context *Data reporting mode*, the purpose of this algorithm is to modify the context *inter-arrival time* I of those non-critical messages by increasing it, so they are not transmitted so often. Consequently, the machine can save transmitting and listening power and thus reduce the energy consumption.

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

$$E_{total} = \sum_{i=1}^{N} E_i \tag{3.7}$$

where N the total number of nodes. The energy consumed by each node can be calculated as in 3.3.

3.2.3 Algorithm 3: Transmission of Relevant Data

This algorithm targets regular monitoring applications, which are the most common types of applications in M2M communications. Regular monitoring applications have low QoS requirements: no real time, priority or accuracy in the data is demanded. In addition, since they monitor certain variable periodically they fall into the time-driven applications category.

For such applications, energy can be saved by applying a data filter, defining a set of threshold values so only relevant data is transmitted by the MTC devices. This will change the context *Data reporting mode* so the machines behave as if they were running a query-driven application, transmitting data only when the criteria of the queries are met.

The *Data filter* defines numeric threshold values based on the *data source*, which trigger transmissions on the MTC device [34]:

- Hard Threshold H_T: once the data source value reaches this threshold, the MTC device switches on the radio module to transmit the sensed data to the MTC server.
- Sampling Interval S_T: it represents small variations in the data source, triggering the MTC 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 MTC device. In case the data source does not meet H_T nor S_T for a time equal or greater than T_R, the MTC device transmits the last sensed value whether it satisfies the thresholds or not. This policy prevents the MTC 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 it will pass until the MTC device demands a new context modification request.

The machine handles the *Data source* by sensing the data source periodically. Once the sensed value reaches the hard threshold, the machine switches on the radio module to transmit the sensed data to the MTC server. The last transmitted value is stored in memory for future reference in a variable called V. The next time the data source is sensed, it is compared with V and transmitted only if the following conditions are met:

- The new value is greater or equal than the hard threshold H_T.
- The new value differs from V by an amount equal or greater than the sampling interval S_T *OR*
- The new value differs from V by an amount equal or less than the sampling interval S_T

Consequently, the machine only transmits data when the data source is in the range of concern. In case the data source is not within the range of concern for a time equal or greater than T_R, the machine switches on the radio module and transmits the last sensed value whether it satisfies the hard threshold and sampling interval or not. This policy prevents the machine to be silent for long periods of time, as well as uncertainty on the network whether the machine is alive or not.

3.2.4 Algorithm 4: Modification of the Average Packet Size

As we mentioned previously, M2M applications are characterized for transmitting small data packets. When a packet transmission is performed, the data measured by the machine is preceded by a header which is used by the eNB for machine and packet identification. If it is possible to merge different data packets by combining them so a single transmission is performed for several packets, the machine would transmit fewer bytes since the same header would correspond to all the combined packets.

In this algorithm, we propose to modify the context *Average packet size* of the machine, combining multiple packets into a single one. Thus, the context *Inter-arrival time* is also modified, if P packets are combined then the inter-arrival time has to be delayed, i.e. increased by P. This leads to wonder how many packets can actually be aggregated? Then the concept of buffer is then introduced. A buffer is part of the context *Hardware*, a region of the physical memory storage used to temporarily store the packets that are going later to be transmitted by the machine. The size of the buffer of the number of packets that can be buffered by the machine can be calculated as the following:

$$B = M_T - M_{OS} \tag{3.8}$$

where M_T represents the total memory of the machine, M_{OS} the memory used by the operative system and B the buffer size, i.e. available memory. The number of packets that can then be buffered by the machine can be formulated as:

$$P = \frac{B}{A} \tag{3.9}$$

where P is the buffered packets, B the buffer and A the default average packet size, which depends on the M2M application.

3.3 Context-Aware Framework

To find a solution that solves the challenges mentioned in section 1.1, that can improve the energy efficiency of a wide range of M2M applications, we propose a context-aware framework that brings dynamic adaptations to MTC devices in response to changes in the contexts observed by the machines. Those observed contexts can be, as discussed in section 3.1: data reporting mode, inter-arrival time, average packet size, QoS features among others.

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

- What contexts should be modified? All that can improve energy efficiency and fits the target application. By fitting is it meant that it is not possible to modified certain contexts if the data reporting mode does not allow it.
- What contexts must be analyzed to make adaptation decisions? All that can help to categorize the machine itself and the way it operates.
- How often adaptations are to be performed?
- How autonomous must be the framework? As much 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 close-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 [35].

To begin with, we consider the contexts that will work as input and output of the framework. Then, we analyze the way those inputs can affect the operation of the machine so suitable outputs can be given to it. Finally, we propose the work flow of the elements of the M2M network: MTC device, MTC server and eNB.

3.3.1 Outputs of the Framework

To define the potential outputs of the framework it is crucial to think in terms of energy efficiency, considering the contexts which can lead to an enhancement on the energy efficiency and extend the operative lifetime of the MTC devices. By following that approach, we analyze table 3.1 so the following contexts are defined as outputs: *Inter-arrival time*, *Average packet size*, *Data filter*, *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.

In addition, by increasing the *Average packet size* of the MTC device by merging multiple packets, it is possible to reduce redundant information in the header of the packet. Increasing the average packet size causes an extension of the inter-arrival time, since it is directly related to the MTC device data transmissions.

The *Data filter* as discussed before reduces the amount of information sent by the MTC device by means of threshold values, decreasing the amount of transmitting power.

Packet Omission can be used both to reduce the amount of packets sent by the machine and to reduce the energy wasted with extra packet headers. This output is regulated by a transmission factor $0 < \alpha \leq 1$, which represents the percentage of packet transmissions. The packets can be omitted because of the inherent correlation among data that has been measured and collected. For example a temperature meter may measure temperature once every hour but the temperature in a whole day may vary slightly. After data compression, many packets can be omitted. Similarly in a cluster where the 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.

The *Cluster ID* and *Cluster head location* are provided by the MTC server if clustering is used, which groups multiple MTC devices together.

3.3.2 Inputs of the Framework

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

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

The *Data reporting mode* let 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 [33]. The time-driven data reporting mode is used by applications that demand periodic data monitoring. In such method the MTC 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 MTC devices react to certain critical event or query sent by the MTC server.

In table 3.2 we depict a binary representation of the existing data reporting modes in M2M communications, including examples of existing applications in the market.

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	-

Table 3.2: Data Reporting Modes in M2M applications.

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 and thus extension of the average packet size and inter-arrival time.

Event-driven applications usually transmit data on very specific scenarios. Normally, applications fall in this category when they use priority alarm messages (PAM), urgent packets that 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 is used. Table 3.3 depicts a summary of the framework outputs supported by each data reporting mode.

	Outputs			
	Packet Omission	Inter-Arrival Time	Average Packet Size	Data Filter
Time-driven	x	x	x	x
Query-driven	x	x	x	x
Event-driven	-	-	-	-

Table 3.3: Summary of outputs supported by different Data Reporting Modes of M2M applications.

In table 3.4 we show a list of possible combinations of *QoS Features* demanded by different M2M applications.

QoS Features			Application examples
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

Table 3.4: QoS Features for M2M applications.

The **Accuracy** requirement relates to the *data source*, which is considered to be accurate if it is free from errors or omissions [36]. Thus, it is possible to omit packets if the application does not require accuracy. On the opposite case, all the packets need to be reported.

When accuracy is not required, it is possible to define a transmission factor $\alpha < 1$. As an example, we have $\alpha = 0.7$ meaning that 70% of the packets are transmitted, while the remaining 30% are being omitted. On the other hand, when accuracy is demanded no packets can be omitted so the transmission factor would be $\alpha = 1$, i.e. all packets are reported. Omitting packet transmissions leads to a 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 are generated by a random event no inter-arrival time nor average packet size modifications are supported. Nevertheless, such messages could be triggered by thresholds defined by the data filter.

The **Real Time** feature is demanded by applications that need a constant active connection with the network, being surveillance applications a good example. 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.

A summary of the outputs supported by M2M applications based on their QoS features is described in table 3.5.

	Outputs			
	Packet Omission	Inter-Arrival Time	Average Packet Size	Data Filter
Real Time	x	-	-	-
Priority	-	-	-	x
Accuracy	x	x	x	-

Table 3.5: Summary of outputs supported by different QoS Features.

Finally, we study the *Connectivity* of the machines. If clustering is used, the MTC device will then stop communicating directly to the eNB and instead do it to the CH. The MTC device, which becomes a cluster member, requires a lower transmitting power since the CH is in the near neighborhood.

3.3.3 Design of 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 MTC device, MTC server and eNB, which flowcharts are depicted in figure 3.2, 3.3 and 3.4 respectively.

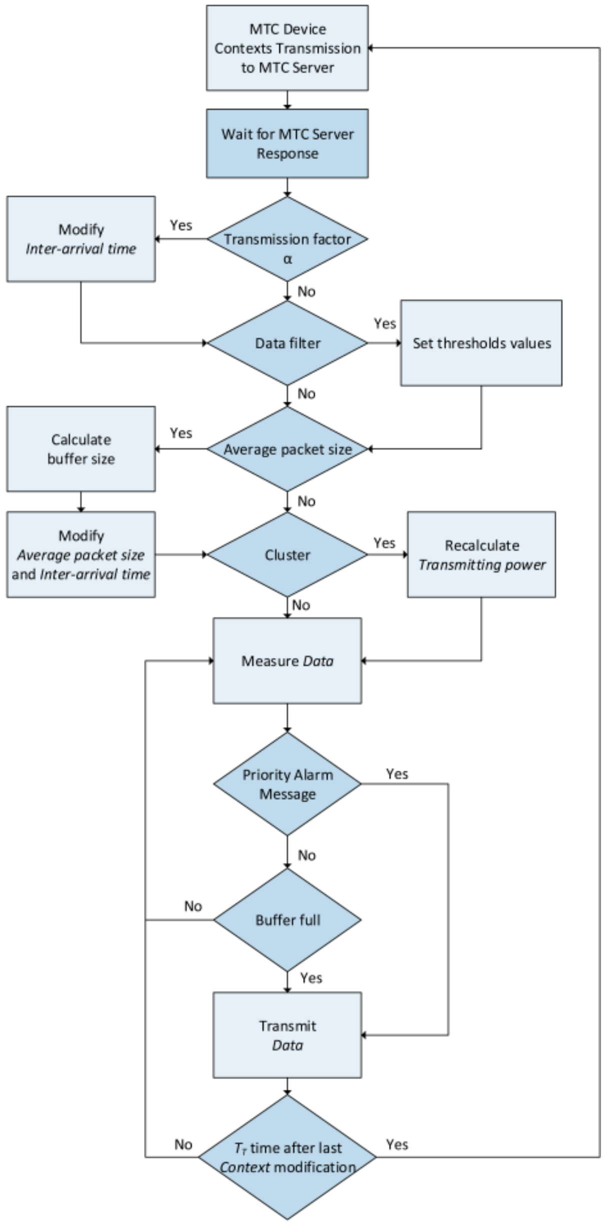

Figure 3.2: Flowchart of the MTC Device.

We begin explaining the work flow of the **MTC device**, which first step is to send a series of contexts to the MTC server, including its *Machine ID*, *Data reporting mode* and *Location*. Once the MTC server replies back the context modification request, the MTC device analyzes the response by first studying if there is any transmission factor. Thus, the *inter-arrival time* will be modified as the following expression:

$$I_N = \frac{I}{\alpha} \tag{3.10}$$

where I_N is the new *inter-arrival time*, I is the default *inter-arrival time* of the machine and α the transmission factor computed by the MTC server. Then, the MTC device verifies if there is a data filter to be set. If so, the machine set the proper variables for 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 this operation by calculating the buffer size B and the number of packets P to be combined, as expressed in formula 3.8 and 3.9 respectively.

Then, the machine modifies its *average packet size* to fit as much packets as possible in its buffer, which can be formulated as:

$$A_N = A \times P \tag{3.11}$$

$$= A \times \frac{B}{A} \tag{3.12}$$

$$= B \tag{3.13}$$

where the new average packet size A_N is a product of the default average packet size A and the buffered packets P. As it can be seen in the derivation, the new average packet size equals the buffer size. Increasing the average packet size by combining multiple packets allows to reduce the amount of redundant information used in the header of each packet transmission.

The context *Inter-arrival time* is also modified in accordance to the amount of buffered packets, if P packets are combined then the inter-arrival time has to be delayed, i.e. increased by P. Thus, the *inter-arrival time* can be reformulated as:

$$I_N = \frac{I}{\alpha} \times P \tag{3.14}$$

$$= \frac{I}{\alpha} \times \frac{B}{A} \tag{3.15}$$

Note that in formula 3.14 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.

Finally, the MTC device checks the existence of clusters indication sent by the MTC server. If there is a cluster it can attach to, the machine recalculates its transmission power lowering it just enough to communicate with the close-distanced CH, based on the CH location which is sent by the MTC server, as described in algorithm 1.

Later, the MTC device is ready to operate after modifying the different contexts, measuring the data source and storing it in its buffer if it is not a PAM. Once the buffer is full, the machine builds a packet and transmits it to the MTC server. Then, the machine verifies if T_T has expired. If not, the machine continues its regular operation, in the loop of measuring-buffering-transmitting. On the other hand, if T_T has expired then it is time to request another context modification, repeating the whole process again.

The **MTC server** has the task of analyzing the MTC devices contexts and responding a combination of suitable context modifications based on the rules defined before.

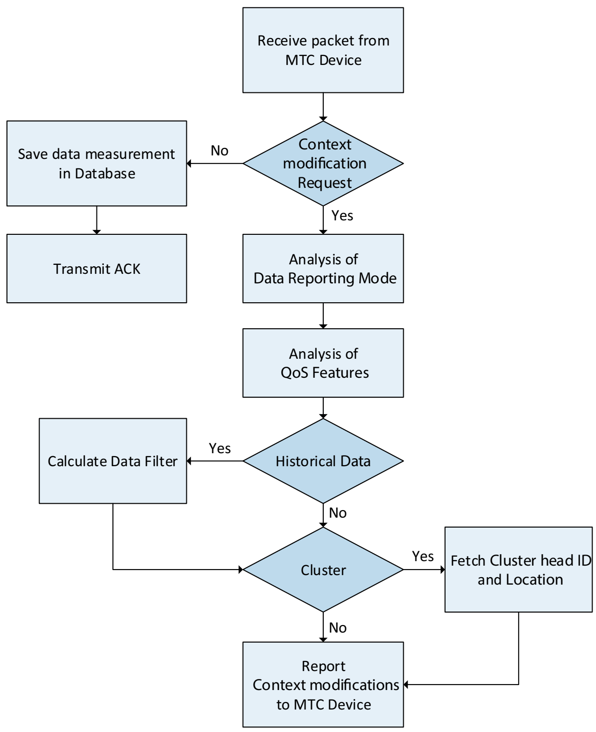

Figure 3.3: Flowchart of the MTC Server.

When the MTC server receives a packet, it analyses if it is a context modification request or if it is just a data report. If the latter, then it stores the measurement in the data base and transmits an ACK message to the MTC device. If the received packet is a context modification request, then it studies the *Data reporting mode* context according to the table 3.2, setting specific flags to the outputs or settings which are supported by the application running in the MTC device.

Then, it proceeds to analyze the *QoS features* context of the machine, according to table 3.4. Based on the rules defined before, it calculates the transmission factor α for the accuracy demanded by the M2M application. It then sets another group of flags for the rest of the outputs which are supported by the machine according to its *QoS features*. Those features that are supported based on both the *Data reporting mode* and *QoS features* are going to be calculated locally by the machine, such as the *inter-arrival time* and the *average packet size*.

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 3.3.1.

Finally, based on the *location* of the MTC device, the MTC server verifies if there is a cluster formed in the neighborhood of the machine. If that is the case, the MTC server includes the *cluster head ID* and *cluster head location* so the machine can recalculate its transmitting power.

To conclude with the operation, the MTC server transmits all this information to the MTC device.

The other agent involved in the context-aware framework is the **eNB** or base station. It basically works as a gateway, redirecting the packets received from the MTC device to the MTC server and vice versa.

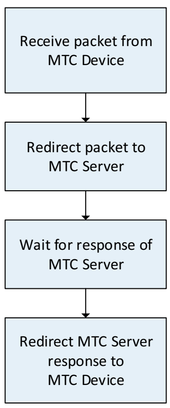

Figure 3.4: Flowchart of the eNB.

Chapter 4

Simulations and Results

This chapter describes the simulations performed for the preliminary algorithms and the proposed context-aware framework for the design of energy-efficient M2M communications. We obtained numerical results of the proposed context-aware framework in Matlab using Monte Carlo simulations, modeling different random traffic and channel effects for the MTC devices in each iteration. The performance metrics include *Average Energy Consumption* in mJ, *Energy Savings* in %, *Residual Energy* in mJ and *Operative Lifetime* in $units$.

We used non-persistent CSMA as MAC protocol and a traffic model that follows a Poisson distribution. We based our simulations in 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. When applicable, packet omissions were performed with a transmission factor $\alpha = 0.5$. PAM were 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, were calculated for a time $T_T = 24$ hours, i.e. new H_T and S_T values were calculated every day based on the measurements of the day before. We calculated the hard threshold H_T as the average of the maximum and minimum values reported on the last T_T, while the S_T was calculated as the standard deviation σ of previous measurements. The T_R was arbitrarily set to 3 times the default inter-arrival time of the respective application.

A few reasonable assumptions were adopted for simplicity:

1. The MTC devices are uniformly distributed within the cell. The BS is located in the center of the cell.

2. All the MTC devices are stationary. The location of all the MTC devices and BS in the network is known.
3. All the MTC devices have enough energy to communicate directly to the BS.
4. The MTC devices can use power control to adjust the transmitting power based on the distance from the receiver.
5. All MTC devices have the same capacities, i.e. are homogeneous.
6. The traffic follows a Poisson Distribution, i.e. the inter-arrival time is exponentially distributed.
7. The application running on all the MTC devices of the network is the same.

Additional simulation parameters, which were chosen according to the LTE uplink budget currently used for MTC [2], are shown in table 4.1.

Number of cells	1
Number of eNB	1
Number of nodes	23
Number of CH	3
eNB height	15 m
eNB 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)

Table 4.1: Simulation parameters.

4.1 Algorithm 1

This algorithm proposes a recalculation of transmitting power of the machines based on the existence of clusters. In this scenario we compare the average energy consumption of the MTC devices when communicating directly to the eNB versus communicating with the eNB through the CHs. The cluster formation algorithm is out of the scope of this thesis, it was handled by a fellow student.

For simplicity and without losing generality, it is studied a set of 3 clusters with 10, 7 and 6 nodes respectively, which is a reasonable number of machines for a cluster and it was generated with the cluster formation routine. During a 1 week simulation-time period, the MTC devices run the

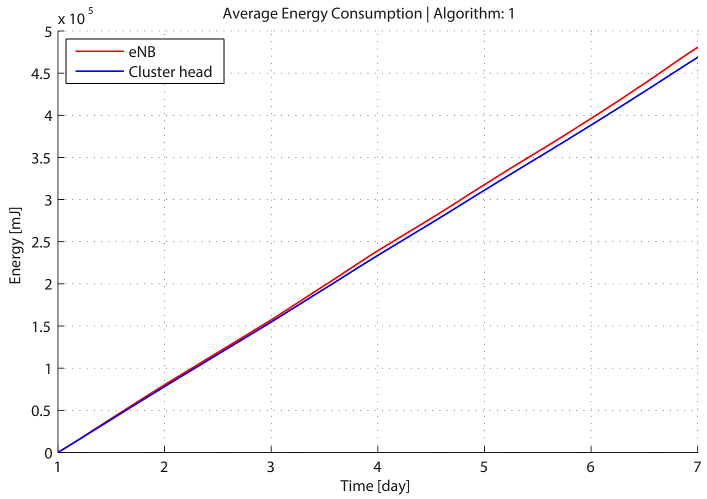

Figure 4.1: Average Energy Consumption implementing Algorithm 1.

application *telehealth*.

It can be seen in figure 4.1 that grouping machines into clusters does not necessarily implies energy saving. Although the cluster members are saving lots of energy by transmitting their data in a short range (i.e. to the CH instead of the eNB) the CH at the same time needs to forward all those packets to the eNB. Machines are grouped into clusters based on location, so the CH uses almost the same power to transmit data to the eNB as the macines in the cluster. Without any data processing small or no energy savings are achieved by grouping nodes into clusters.

Considering the results, we suggest to adopt data aggregation so energy savings can be achieved. Since the data within the cluster is likely to be highly correlated as different authors suggest [37, 38, 39], the CH can combine the data from the cluster members to avoid redundant information. Consequently, only *effective data* is actually transmitted from the CH to the base station.

Based on this finding, another simulation of this algorithm was performed introducing the transmission factor α described in section 3.3.1. This factor has a direct impact on the transmitting power, the less packets are transmitted the less transmitting power is consumed by the CH. At the same time since the CH listens to the channel before transmitting any data, i.e. it consumes listening power every time data is going to be transmitted, the transmission factor also affects the consumption of listening power. Thus, the total energy consumption of the CH can be rewritten as:

$$E = \alpha(P_{tx}t_{tx} + P_l t_l) + P_s t_s \quad (4.1)$$

where α is the transmission factor which is varied in the simulations from 0.1 to 0.5. Figure 4.2 describes the results of using packet omissions when in presence of clustering.

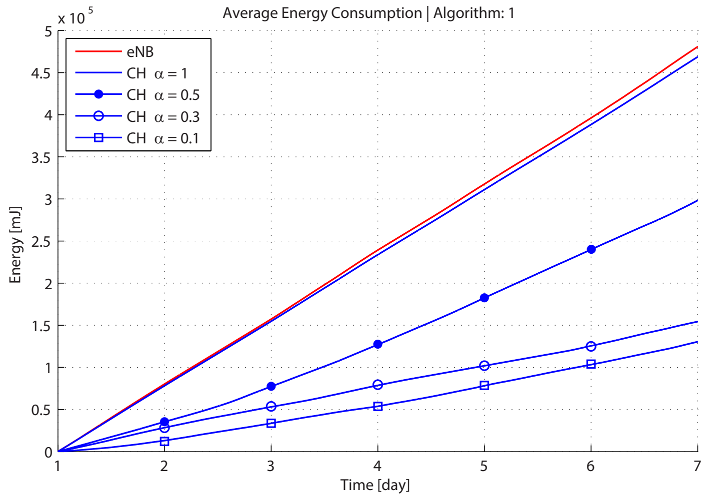

Figure 4.2: Average Energy Consumption introducing the transmission factor α in Algorithm 1.

In figure 4.2 we can see the effect of the transmission factor α when clusters are used in M2M communications. As it is expected, the smaller the value of alpha the lower the total energy consumption of the system, obtaining energy savings of 47.21 %, 66.61 % and 78.26 % for α values of 0.5, 0.3 and 0.1 respectively.

Thus, with this simulation we conclude that clustering can be successfully applied in M2M communications not only for policing, addressing or charging but also to extend the lifetime of the network by means of packet omissions, which reduces data redundancy and allows to save energy by decreasing the transmitting and listening power of the CH. Nevertheless, it has to be considered the fact that the tasks executed by the CH consume more energy than the ones of a non-CH node. This means that the battery life of the CH is shorter than a non-CH node, and when it gets drained the rest of the nodes will lose communication with the base station. Considering this, novel techniques need to be applied so the task of the CH is "rotated" within the members of a cluster group in a smart way so the energy consumption of the system is distributed among all the nodes and thus, the operating lifetime of the network can be maximized. Related work about load balancing in clusters can be found in [38, 39, 40].

4.2 Algorithm 2

This algorithm introduces an adjustment of the inter-arrival time for applications that have a hybrid data reporting method, where messages containing no critical information are transmitted periodically, which clearly represents a waste of energy. In this scenario the nodes communicate directly to the eNB and no clusters are formed. We compare the average energy consumption of 2 cases: when the nodes transmit non-critical data packets with the default inter-arrival time (i.e. 600 seconds for the *home security* application) and when the nodes transmit only one non-critical data packet a day. For this specific application, $3600 \times 24 \div 600 = 144$ non-critical data packets are sent a day per node with the default inter-arrival time (excluding priority alarms) with the hybrid data reporting method.

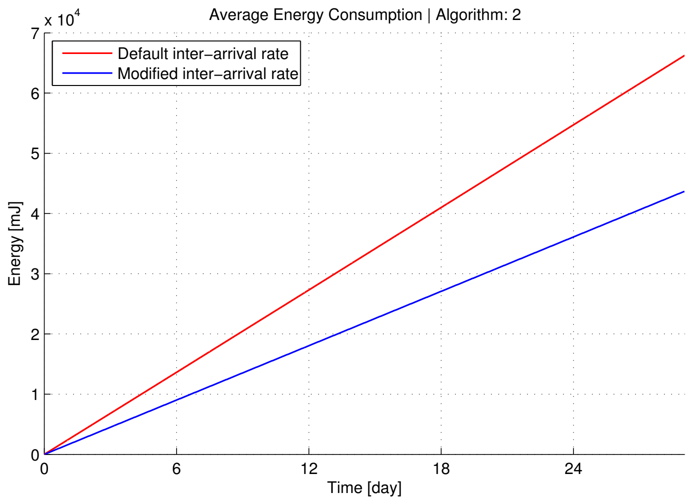

Figure 4.3: Average Energy Consumption implementing Algorithm 2.

Figure 4.3 shows the difference in terms of energy consumption when the inter-arrival rate is modified of M2M applications with hybrid data report methods, which behave as an event-driven and time-driven application at the same time. By prolonging the inter-arrival time of 600 seconds to 1 day of those non-critical packets in the *home security* application, energy savings of 34% can be achieved. It is also concluded that there is a trade-off between the transmitting and sleeping power; the longer the nodes are set to sleep the less transmitting power is used, but at the same time the sleeping power is increased. Nevertheless, the simulation proves that is a reasonable trade-off to extend the operative lifetime of the network.

4.3 Algorithm 3

This algorithm suggests the usage of a data filter so only relevant information is reported by the machines. The algorithm is applied to real data measurements of a climate application for the parameter *temperature* of a weather station located in Malmö - Sweden captured during the month of June 2013 [41]. The measurements were taken 3 times an hour for an average inter-arrival time of $3 \div 60 = 20$ minutes.

Figure 4.4 shows the average energy consumption of this case study. By implementing the data filter on the *temperature* measured by a climate application, a total of 1583 packets were transmitted in comparison to 2144 when no data filter is applied, which represents 73.83 % of all the data and decreasing the energy consumption by 22.01 %. Smaller values of the hard, soft and time thresholds provide a more accurate representation of the network at the cost of more energy consumption, which is the trade-off of this algorithm.

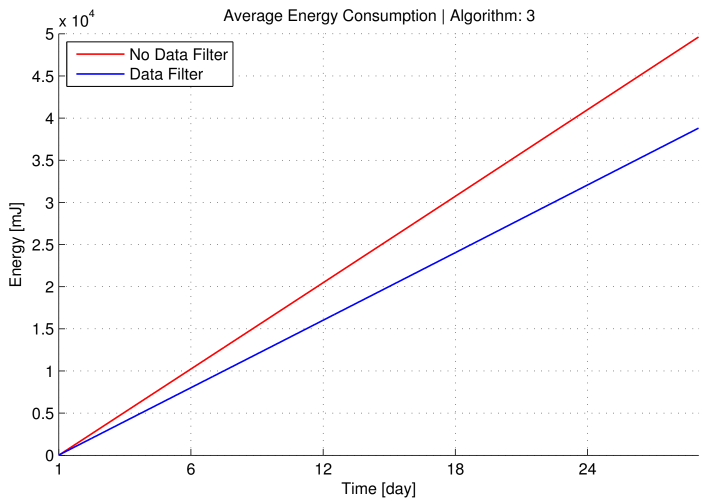

Figure 4.4: Average Energy Consumption implementing Algorithm 3.

It is important to notice that the settings of the thresholds depend on the *data reporting mode* and the urgency of the parameters sensed by the nodes. The lower the accuracy on the data demanded by the user the higher the soft and time thresholds can be, which may provide higher energy efficiency.

4.4 Algorithm 4

The idea of this algorithm is to modify the *Average Packet Size*, combining several data packets so that fewer transmissions are done, extending the *inter-arrival time* and decreasing redundant header information. To simulate this algorithm it was used the telehealth application. The buffer size was set to 5000 bytes, which according to formula 3.9: $5000 \div 128 = 39$ packets can be aggregated by the machine.

It can be appreciated in figure 4.5 how the modification of *average packet size* can influence the energy efficiency on the network, multiple packets can be aggregated in a bigger one which contributes to header reduction, providing energy savings of 21% in comparison to the default average packet size.

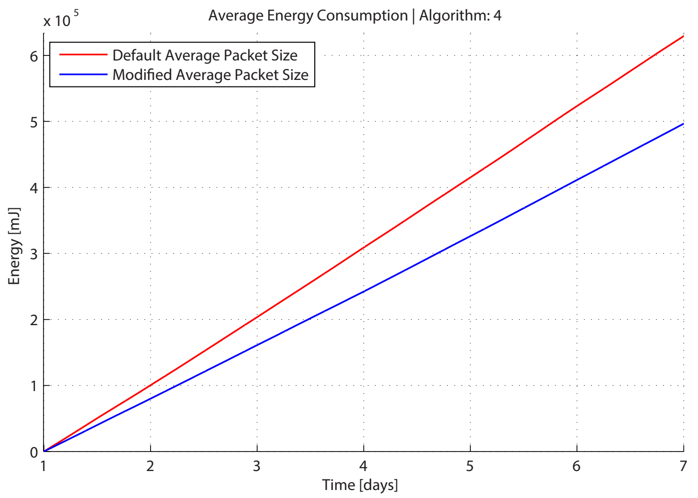

Figure 4.5: Average Energy Consumption implementing Algorithm 4.

4.5 Summary of Preliminary Results

As we could see in the previous sections, we obtained successful results when testing each energy-efficient algorithm individually for the scenarios described above. Different conclusions were obtained, which were important for determining the way those algorithms are combined in the proposed context-aware framework, which results are shown in the next section.

The following table shows a summary of the preliminary results.

	Context Modifications	Energy Savings
Algorithm 1	Connectivity, Transmitting Power, Inter-Arrival Time	78.26 %
Algorithm 2	Inter-Arrival Time	34 %
Algorithm 3	Data filter	22.01 %
Algorithm 4	Inter-Arrival Time, Average Packet Size	21 %

Table 4.2: Summary of Preliminary Results.

4.6 Context-Aware Framework

After obtaining preliminary results by simulating each algorithm separately, we proceeded simulating the context-aware framework according to the analyses studied in previous sections. We performed simulations using four major M2M applications: *climate, telehealth, home security* and *smart metering*, which average packet size and inter-arrival time can be found in table 4.1.

In addition of the Average Energy Consumption and Energy Savings calculated for the different algorithms, it was also computed the Residual Energy and Operative Lifetime Extension for each application in order to verify the efficiency of the framework. Based on real data measurements [41, 42] it was possible to study the usage of the data filter for both the climate and telehealth applications, for which different sampling intervals S_T were used to investigate the impact of it on the performance metrics.

In figure 4.6 we depict the average energy consumption for each application, while in figure 4.7 the energy savings achieved by the context-aware framework. Figure 4.8 depicts the operative lifetime extension factor for different conditions.

A linear energy consumption it is evident in figure 4.6 for the climate application due to periodic data transmissions, which characterize regular monitoring applications. As it is shown in figure 4.7, the proposed context-aware framework obtained 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.

It is also noticeable in figure 4.6 the influence of S_T on the average energy consumption, the smaller the S_T the higher the energy consumption since the MTC 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. A small difference in terms of

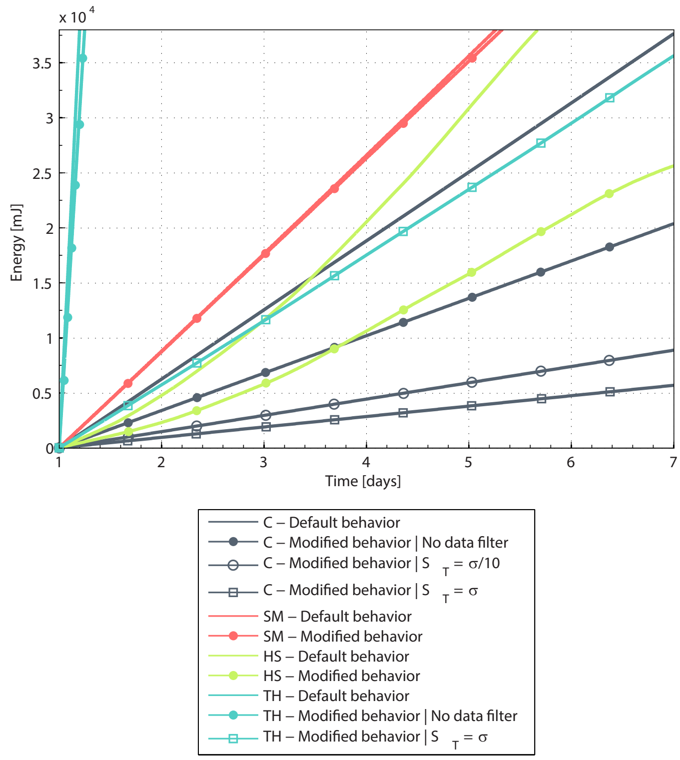

Figure 4.6: Average Energy Consumption - Context-Aware Framework.

energy saving can have a significant impact on the operative lifetime of the machines, obtaining an operative lifetime extension of 1.85 when no data filter is used while 3.8, 5.11 and 6.5 times when the S_T is set to $\sigma/10$, $\sigma/5$ and σ respectively.

In the case study of the **telehealth application** we analyzed the *heart rate* parameter from the data set #12 of vital signs of the University of Queensland [42]. Figure 4.6 depicts the big contrast on the average energy consumption with and without data filtering.

As it can be seen in figure 4.7 we accomplished positive results, achieving energy savings of 19% with no data filter while a dramatic 96% when introducing thresholds values, corresponding to a lifetime extension factor of up to 32 times the default value as shown in figure 4.8.

The reason of obtaining lower energy savings in comparison to the climate application when considering the modified behavior without data filter lies on the *QoS features*, the telehealth application requires accuracy while the climate application does not. Thus, the transmission factor α is set to

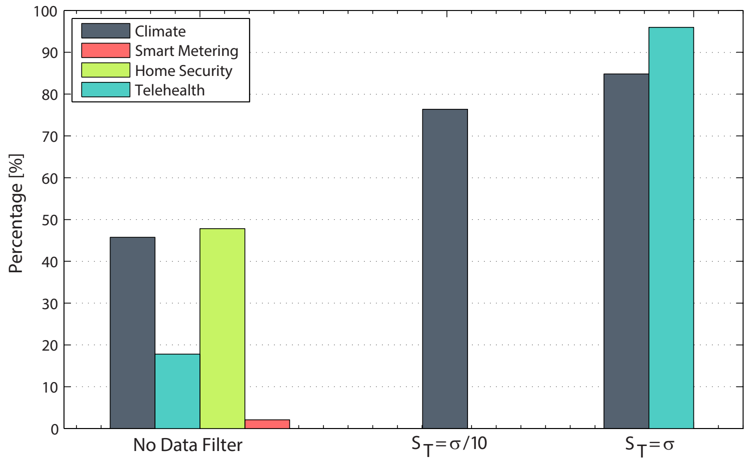

Figure 4.7: Energy Savings - Context-Aware Framework.

1 so no packets are omitted, even when they are redundant.

In addition, the dramatic energy savings and operative lifetime extension obtained by the framework when data filtering is performed has to do with the 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.

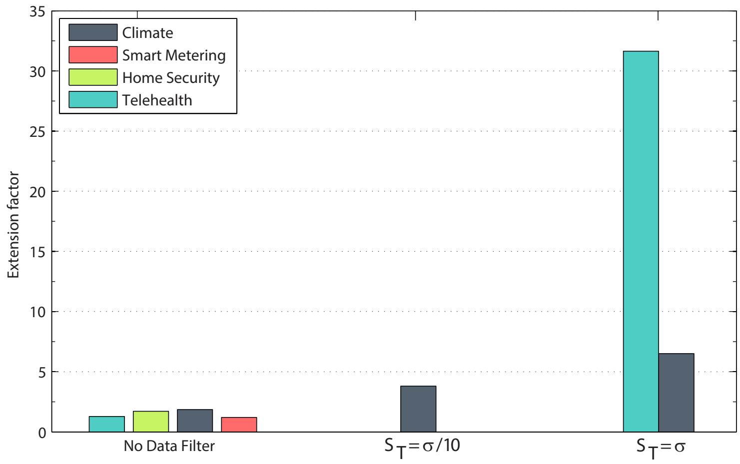

Figure 4.8: Operative Lifetime Extension - Context-Aware Framework.

Finally, for the **smart metering** and **home security applications** we achieved energy savings of 2.5% and 47% respectively, as seen in 4.7. 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 have a significant impact in 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 no meaningful energy savings nor operative lifetime extension were achieved in this case, as seen in figure 4.6.

On the other hand, 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 that correspond to an operative lifetime extension of 1.75 times, which matches the result obtained by the time-driven climate application when no data filter is used, confirming the efficiency of the context-aware framework for M2M communications.

Chapter 5

Conclusions

This chapter completes this project condensing the content in a summary, findings and future work.

5.1 Summary

The market of M2M communications in cellular networks is expanding rapidly, making the IoT a reality. It is estimated that 50 billion MTC devices are going to be connected by 2020 and although hard work is being done to standardize the way this communications operate, there are still many aspects that need to be improved in this field. At present, different vendors use their own protocols which makes difficult to find adaptive solutions.

Among the matters that need to be addressed in M2M communications, the energy efficiency issue a critical one which was the main aim of this project. To improve the energy efficiency in M2M communications is important to consider that MTC devices are normally powered by batteries, intelligent algorithms are needed to regulate the way the power is consumed by the machines in order to maximize the operative lifetime of the M2M network.

The way we addressed this challenge in this project was to adopt context and context-aware concepts investigated in computing systems to M2M communications. We proposed that a context can be any information that can be used to characterize the state of a MTC device, and based on that idea we defined a set of contexts, going beyond location to exploit different features of the MTC devices such as *data reporting mode*, *inter-arrival time* and *average packet size* in search of improving the energy efficiency of the network.

In the early stages of the investigation, different algorithms targeting different scenarios and types of applications were proposed and individually tested to confirm the veracity of the proposed contexts and algorithms themselves, obtaining satisfactory results. Then, the context-aware framework was formulated with the idea of supporting a wide range of M2M applications, no matter the difference in their contexts.

This way, it was possible to make the MTC devices adapt automatically their operative modes without human intervention, obtaining energy savings and extending their operative lifetime.

5.2 Findings

1. It was clearly seen with Algorithm 1 that clustering does not necessarily mean energy savings, as it was initially thought, since it is possible that the transmitting power used in the MTCD-CH-eNB link is higher than the power used by the MTCD-eNB link. Data aggregation might be needed to achieve energy savings.
2. When extending the inter-arrival time there is a trade-off between the transmitting and sleeping power used by the machine. The longer the inter-arrival time the less the transmitting power used by the MTC devices, while the sleeping power is increased. Nevertheless, the sleeping power is much lower than the transmitting power and the simulations prove that it is a reasonable trade-off to extend the operate life of the network.
3. Implementing a data filter on the data source improves the energy efficiency of the MTC device. The sampling interval S_T determines how accurate the picture of the network is going to be, since it is responsible for triggering data transmissions once the hard threshold H_T is reached. The bigger the S_T the higher the energy efficiency. Is up to the MTC server how to set this parameter based on its needs.
4. The buffer size of the MTC devices plays and important roll when combining packets. A larger buffer size provides higher energy efficiency, since more packets can be combined into one and the header reduction is bigger.
5. The system-level simulations of the context-aware framework performed with four major M2M applications: Climate, Telehealth, Home Security and Smart Metering show how effective and responsive is the proposed context-aware framework for handling different types of applications, demonstrating that the usage of contexts in M2M communications is indeed beneficial, offering considerable energy savings of up to 96% and prolonging up to 31.6 times the operating lifetime of the MTC devices in the best case scenario.

5.3 Future Work

The achievements obtained in this master's degree project provide new perspectives about M2M communications that need to be investigated. The proposed context-aware framework for M2M communications has certain limitations, for example it does not consider mobile MTC devices, which would certainly bring up new challenges to solve such as handover or inter-cell interference.

In addition, no real time communications are analyzed when studying the *QoS features* - i.e. applications such as surveillance are not supported by the proposed context-aware framework. The IoT is evolving fast and new features and demands will arise in the following years, which will open a range of new possibilities that will need to be studied.

No H2H communications are considered in this framework, which would coexist with M2M communications when using cellular networks. Thus, this framework could be extended with such functionality, which probably would require defining additional contexts.

Bibliography

[1] A.-E. S. Al-Sanad, "[phi]-mac: An energy-aware multi-phase mac protocol for wireless sensor networks," Master's thesis, University of Victoria, 2005.

[2] 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.

[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] I. Juniper Networks, "Machine-to-machine (m2m) - the rise of the machines," 2011.

[5] Ericsson, "Operators need an ecosystem to support 50 billion connections," 2010.

[6] 3GPP TS 22.368 V12.1.0, "Service requirements for Machine-Type Communications (MTC)," tech. rep., December 2012.

[7] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, "A first look at cellular machine-to-machine trafic - large scale measurement and characterization," 2012.

[8] S.-Y. Lien and K.-C. Chen, "Massive access management for QoS guarantees in 3GPP machine-to-machine communications," vol. 15, no. 3, pp. 311–313, 2011.

[9] S.-Y. Lien, K.-C. Chen, and Y. Lin, "Toward ubiquitous massive accesses in 3GPP machine-to-machine communications," vol. 49, no. 4, pp. 66–74, 2011.

[10] K. Zheng, F. Hu, W. Wang, W. Xiang, and M. Dohler, "Radio resource allocation in LTE-advanced cellular networks with M2M communications," vol. 50, no. 7, pp. 184–192, 2012.

[11] S. Y. Shin and D. Triwicaksono, “Radio resource control scheme for machine-to-machine communication in LTE infrastructure,” in *ICT Convergence (ICTC), 2012 International Conference on*, pp. 1–6, 2012.

[12] A. S. Lioumpas and A. Alexiou, “Uplink scheduling for machine-to-machine communications in LTE-based cellular systems,” in *GLOBECOM Workshops (GC Wkshps), 2011 IEEE*, pp. 353–357, 2011.

[13] A. G. Gotsis, A. S. Lioumpas, and A. Alexiou, “M2M scheduling over LTE: Challenges and new perspectives,” vol. 7, no. 3, pp. 34–39, 2012.

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

[15] A. Schmidt, M. Beigl, and H. w. Gellersen, “There is more to context than location,” 1998.

[16] Y. Oh, A. Schmidt, and W. Woo, “Designing, developing, and evaluating context-aware systems,” in *Multimedia and Ubiquitous Engineering, 2007. MUE ’07. International Conference on*, pp. 1158–1163, 2007.

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

[18] I. W. Hailiang Mei, “Context-aware optimal assignment of a chain-like processing task onto chain-like resources in m-health.”

[19] J. gu Song, S. Kim, and G. cheol Park, “Towards a mobile and context-aware technology based healthcare system,” in *Frontiers in the Convergence of Bioscience and Information Technologies, 2007. FBIT 2007*, pp. 591–594, 2007.

[20] L. Bedogni, M. Di Felice, and L. Bononi, “By train or by car? detecting the user’s motion type through smartphone sensors data,” in *Wireless Days (WD), 2012 IFIP*, pp. 1–6, 2012.

[21] 3GPP TR 23.888 V11.0.0, “System Improvements for Machine-Type Communications (MTC),” tech. rep., September 2012.

[22] EnerNOC Inc, “What is m2m communications,” February 2013.

[23] Bob Emmerson, “M2M: the Internet of 50 billion devices,” January 2010.

[24] Dalicia Bouallouche, “Congestion control in the context of machine type communication in long term evolution networks: a dynamic load balancing approach,” Master’s thesis, University of Rennes 1, 2012.

[25] A. Maeder, P. Rost, and Staehle, "The Challenges of M2M Communications for the Cellular Radio Access Network," in *EuroView 2011*, 2011.

[26] R. Huang, *IEEE 802.16p Machine to Machine (M2M) Evaluation Methodology Document (EMD)*, May 2011.

[27] F. Gebali, *Computer Communication Networks Analysis and Design*. Northstar Digital Design Inc, 2002.

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

[29] P. J. Brown, J. D. Bovey, and X. Chen, "Context-aware applications: from the laboratory to the marketplace," *IEEE Pers Commun*, vol. 4, no. 5, pp. 58–64, 1997.

[30] 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.

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

[32] B. Schilit, N. Adams, and R. Want, "Context-aware computing applications," in *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*, pp. 85–90, 1994.

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

[34] A. Manjeshwar and D. P. Agrawal, "Apteen: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless," in *Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM*, 2002.

[35] 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.

[36] K. O'Connor, "What does complete appropriate and accurate mean?," June 2013.

[37] M. Ye, C. Li, G. Chen, and J. Wu, "Eecs: an energy efficient clustering scheme in wireless sensor networks," in *Performance, Computing, and*

Communications Conference, 2005. IPCCC 2005. 24th IEEE International, pp. 535–540, 2005.

[38] 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.

[39] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” vol. 1, no. 4, pp. 660–670, 2002.

[40] Y. Ossama and S. Fahmy, “Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks,” vol. 3, no. 4, pp. 366–379, 2004.

[41] NOAA’s National Climatic Data Center, “NNDC Climate Data Online,” June 2013.

[42] The University of Queensland, “The university of queensland vital signs dataset,” June 2013.