Online Estimation of Battery Lifetime for Wireless Sensor Network

Emmanuel Nataf, Olivier Festor

Online Estimation of Battery Lifetime for Wireless Sensor Network

Emmanuel Nataf, Olivier Festor

Online Estimation of Battery Lifetime for Wireless Sensor Network

Emmanuel Nataf*, Olivier Festor

Project-Teams Madynes

Abstract: Battery is a major hardware component of wireless sensor networks. Most of them have no power supply and are generally deployed for a long time. Researches have been done on battery physical model and their adaptation for sensors. We present an implementation on a real sensor operating system and how architectural constraints have been assumed. Experiments have been made in order to test the impact of some parameter, as the application throughput, on the battery lifetime.

Key-words: battery, wireless sensors network, Contiki, Cooja

* Université de Lorraine

Éstimation en ligne de la Durée de Vie des Batteries pour les Réseaux de Capteurs sans Fil

Résumé : La batterie est un composant matériel majeur dans les réseaux de capteurs sans fil. Des recherches ont été menées sur les modèles physique des batteries et leur adaptation pour les capteurs. Nous présentons une implantation sur un véritable système d'exploitation de capteurs et comment les contraintes d'architecture ont pu être pris en compte. Des expériences ont ensuite été menées pour tester l'influence de certains paramètres comme le débit d'application sur la durée de vie des batteries

Mots-clés : batterie, réseaux de capteurs sans fil, Contiki, Cooja

1 Introduction

Wireless Sensors Networks have a growing place in industrial and domestical domains. Generally sensors need to be small, cheap and easy to deploy on any physical environment. These constrained networks are made possible by the processor size reduction and high performance battery. Nevertheless these networks have to be care of their energy consumption because they are usually deployed for a long time or because sensors are so small that their battery has a limited capacity. The growing interest on battery lifetime estimation for WSN appears in the standardization work on these network [6]. They defines a routing protocol with one metric being the remaining battery level [20].

Battery models that relate its use with its lifetime have been proposed since embedded systems (laptop, cellular phone) are widespread. All of them take into account two phenomenons occuring in battery cell; the *Rate Capacity Effect* and the *Recovery Effect* [9]. The first gives the energy consumed under a constant current load, when transmitting for example, and the second gives the energy recovered during inactivity or low current load. At a constant current load, oxidation at the anode induces reduction at the cathode. The reduction decreases the concentration of positive ions near the cathode and so the available energy. But during inactivity or low current load the positive ions near the anode have time to move toward the cathode, thus increasing the available energy and so the battery lifetime. A stochastic model is given in [9], but the more accurate modeling is given by the analytical equation (1) [13, 15, 16]. α denotes the total battery capacity that is equals to the current load $i(\tau)$ consumed since the use of the battery until the time L (the battery *Lifetime*) and the current that was unavailable because positive ion concentration was not sufficient at the cathode. The β parameter is the diffusion coefficient of electroactive species inside the battery.

$$\alpha = \int_0^L i(\tau)d\tau + 2\sum_{m=1}^{\infty} \int_0^L i(\tau)e^{-\beta^2 m^2(L-\tau)}d\tau \qquad (1)$$

Use of this model can be found in [14, 18] to schedule tasks for embedded system or to simulation [19]. It is widely used for example by a circuit-based battery model [21] or in WSN routing simulation [17]. The model can not be implemented as is on WSN nodes albeit some results achieved using the first part of equation (1) [5, 7]. In so doing, they do not take into account the recovery process that occurs during idle time despite it is generally up to 90% of the WSN node timelife.

The contribution of [12] is an approximation of the equation (1) that can be implemented on WSN node. Its global equation (2) recursively computes $\sigma(L_n)$ that is the consumed charge in *mA-min* at time L_n. This computation depends on the consumed charge at time L_{n-1}, the time interval $\Delta = L_n - L_{n-1}$ being constant.

$$\sigma(L_n) = \sum_{k=1}^{n} I_k\delta_k + \lambda\left(\sigma(L_{n-1}) - \sum_{k=1}^{n-1} I_k\delta_k\right) + 2I_n A(L_n, L_{n-1} + \delta_k, L_{n-1}) \qquad (2)$$

With this approach, one can know the remaining charge in the battery at time L_n by the difference between $\sigma(L_n)$ and the initial battery parameter α of the equation (1). The recovery effect of the battery is computed through the function A of the equation (2) that is approximated by the use of an f function:

$$A(L_n, L_{n-1} + \delta_k, L_{n-1}) = \sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2(\mathbf{L_n - L_{n-1} - \delta_k})} - e^{-\beta^2 m^2(L_n - L_{n-1})}}{\beta^2 m^2} \simeq \frac{f(\nu)}{\beta^2} - \frac{f(\Delta)}{\beta^2} \qquad (3)$$

This approximation depends on the idle time of the mote during the Δ interval, given by the difference $\nu = \Delta - \delta_k$ (ν is the idle time of the L_{n-1} period and δ_k its activity time). Following [12], the f function can be also approximated by the use of the equations (4) to (6) (see [10, 11] for details):

$$\frac{f(\nu)}{\beta^2} = \frac{\nu}{2} - \frac{\sqrt{\pi}}{\beta}.\sqrt{\nu} + \frac{\pi^2}{6\beta^2} \tag{4}$$

$$\sqrt{\nu} \simeq \sqrt{a} + \frac{\nu - a}{2\sqrt{a}} \tag{5}$$

$$\frac{f(\Delta)}{\beta^2} = \sum_{m=1}^{\infty} \frac{e^{-\beta^2 m^2 \Delta}}{\beta^2 m^2} = c_0 \tag{6}$$

The β parameter (physical diffusion coefficient) in the equations (4) and (6) is a constant value that depends on the battery and is computed from several tests of discharge and the least squares estimation method (along with α). The approximation of the equation (5) concerns $\sqrt{\nu}$ by a mean value : a, computed offline. In our case the mean value a is fixed at 90% of Δ, that is the value we measure on several tests. The last approximation of the equation (6) is computed with the m parameter ranging from 1 to 10 because the sum quickly converges.

The recursivity of the model depends on the λ parameter defined as the ratio $\frac{A(L_{n+1},\delta_1,0)}{A(L_n,\delta_1,0)}$ for each n but that can be bounded by the value $e^{-\beta^2\Delta}$ computed off line.

Our contribution in this paper is to fill the last gap between the approximated model and its implementation within an existing operating system for sensors. We use physical nodes to validate our energy consumption retrieval process and observe in simulated world how battery reacts face to network events and configuration parameters. The remainder of this paper is organized as follow. The next section is focalized on the elementary operations of WSN node and their current draw. Details of the implementation are given in the section 3 and first results are shown in the last part. We give some conclusions and plan future works at the end of the document.

2   Current draw

2.1   Linear draw

The first term in equation (2) is the sum of products between current load I_k and time interval δ_k since the begining of the battery livetime (when $k = 1$) until the time L_n. The δ_k time interval is a sub interval of Δ during which the battery was used by the node ($\delta_k \leq \Delta$). To compute this value we have to know which components of the motes use the current along the time and at which current rate.

Mote's current information can be obtained from datasheet document. For example, the table 1 shows the current load for two mote types (Sky or WSN430 motherboard, respectively with CC2420 and CC1100 communication chips). The columns are the usual states of the duty cycle for any mote : CPU, LPM, TX and RX respectively for processor, low power mode, transmitting and receiving. Currents are given in milli-Ampere (mA). The given values for TX and RX are linked to signal power and throughput configuration of the mote.

The left part of figure 1 shows a short time interval of a real WSN430 mote with the Contiki operating system [4] on the senslab testbed [2]. We plot these states by logging start and stop time on mote serial output that is redirected by senslab to a TCP connexion. These times are accessibles by an energy related application programming interface on Contiki. The CPU and

Mote	CPU	LPM	TX	RX
Sky	1.8	0.0545	17.4	18.8
Wsn430	2	0.02	16.1	15.2

Table 1: Motes current draw

LPM are not overlapping themselves and fill all the time duration. These states are related to the mainboard activities and so either the CPU is active or the low power mode is raised. TX and RX states are related to the radio activities and they are neither used every time nor they overlap each otther. One can note that TX mode only occurs during CPU and that RX mode begins during CPU but can spread on LPM.
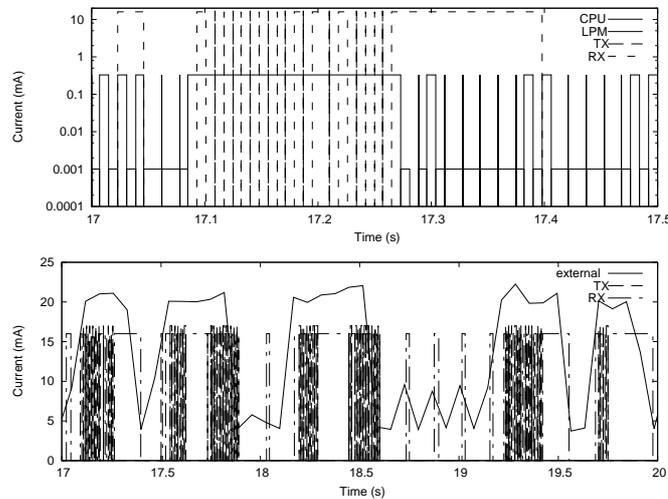

Figure 1: Mote modes and currents

The senslab testbed also provides an external polling of the current consumed by a mote during an experiment. The right part of figure 1 shows the curves of measured current at the mote endpoints and the duties cycles as in the left part (we just leave TX and RX mode that have the main current values). One can see that start and stop times of duties match well with measured current.

Consequently, the current draw of a $\Delta = \delta_{CPU} + \delta_{LPM}$ intervall is the result of the equation (7) where each C_{state} is the current load (cf.table 1) and each δ_{state} is the sum of all state sub-intervals inside Δ.

$$I_k \delta_k = C_{CPU}.\delta_{CPU} + C_{LPM}.\delta_{LPM} + C_{TX}.\delta_{TX} + C_{RX}.\delta_{RX} \tag{7}$$

3 Implementation constraints

Even with the help of [12], we must be care of mote limitations on code size and arithmetics capabilities. The strong constraint is that MSP430 can not handle floating point values but only signed or unsigned integer.

3.1   Offline computation

Approximations given make use of $\sqrt{\pi}$ or π^2 and we had to round them with a new unit as it is show on table 2. Most of them are just a factor of thousand to remove floating point with saving enougth precision. The computed values ν, c_0 and λ are computed with a Δ of $2s$.

	π^2	$\sqrt{\pi}$	β	c_0	λ	a	$\sqrt{a}$	$\frac{1}{2\sqrt{a}}$
R	9.869	1.772	1	1.337	0.967	0.03	0.173	2.886
I	9869	1772	10	1337	967	30	173	2886

Table 2: Real to Integer values

3.2   Online computation

We compute the consumed current every Δ time interval. The values $\delta_{CPU,LPM,TX,RX}$ are provided in milli second and we use the current load of the table 1 with a factor of thousand. The result is in $10^{-3}mA.ms$ and it is converted in $mA.mn$.

We compute ν with the low power mode period and the radio use :

$$\nu = \begin{cases} \delta_{LPM} - (\delta_{TX} + \delta_{RX}) & \text{if } \delta_{LPM} > (\delta_{TX} + \delta_{RX}) \\ 0 & \text{else} \end{cases} \tag{8}$$

If the radio is more used than the CPU is idle then there is no idle time for the battery.

The varying part of the recovery function A (equation 4) is rewritten in order to lose as little precision as possible :

$$\frac{f(\nu)}{\beta^2} = \frac{10^4\beta^2\nu + \pi^2 2.10^3 - 12\beta\sqrt{\pi}\sqrt{\nu}}{\beta^2 12.10^5} \tag{9}$$

At this step we had to use a 64 bits intermediary value to get the result (MSP430 compiler allows such data type but they are actually emulated with several 32 bits values).

Finaly, the remaining energy is computed with the values above and the preceding remaining energy value (a Δ before). We compute the ratio with 255 as 100%. This value is given in RPL routing metric recommandation [20] and we use the present work to implement and test an energy-based routing plane in an other work beyond the scope of this paper. However we compute the remaining energy with a precision of 5 numbers, that is from 255.10^5, so one can observe energy consumption at a very fine coarse.

The code size of the presented implementation is about $12Kb$ on the $50Kb$ allowed by our processor. At the memory level, we save on memory six 32 bits values from one computation to the other (times of CPU, LPM, TX, RX and previous values for $\sigma(L_{n-1})$ and $\sum_{k=1}^{n-1} I_k\delta_k$).

4   Simultation Results

For our simulation we use the Cooja WSN simulator [8] and test a network of nodes build on the sky mote platform. These nodes use IPv6 and the RPL routing protocol to perform a simple collect application during which one or more nodes, the *Senders*, periodically send sensing informations to a receiving node called the *Sink*, configured to be the root of the RPL routing tree. All node have an initial battery charge of $880mAh$. *Senders* send one data packet every second and use the *contikiMac* radio duty cycle [3]. We use a linux *Ubuntu 11.0* desktop computer with a $3GHz$ *Intel* processor and $4Gb$ of memory.

4.1   Booting node

We start our experiment with the observation of the node booting process. The figure 2 puts together the evolution of the battery lifetime and the time spends by activities of the node. During the first minute, on the part (a), one can note a large decrease of the battery[1] to correlate with the strong activity on the part (b) for the same period. Mainly this activity is related to system and network initialization (remains the strong power for TX and RX of the table 1). The recovery of battery is visible at this very close level, during the second minute and the two

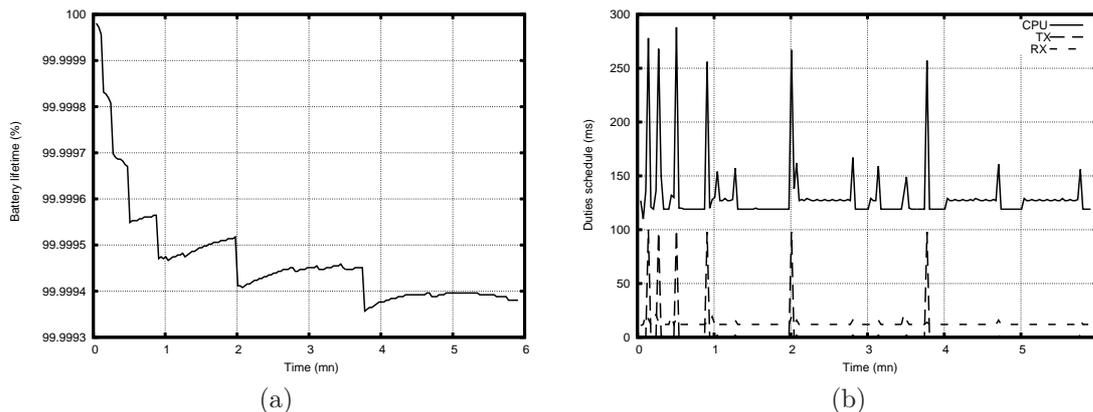(a)   (b)

Figure 2: First minutes of battery lifetime

following ones. These recovery periods match with less activities period.

4.2   Network building

We continue the simulation following the figure 3 (a) where after ten minutes of simulated time, we add five nodes "under" the non-root node to observe the cost of the routing tree building and traffic cumulation. The node 1 is the *Sink* and all other are *Sender*. We leave these nodes sending and forwarding (only for the node 2) data packets during ten minutes and remove the five nodes previously added. This can occurs in real life for example with mobile nodes or with perturbed communication environment. A last point of interest, at the thirty fifth minute, is when a node lost its destination node (its default next hop router) because the network has to build itself again.

The figure 3(b) plots the remaining battery lifetime of the node 2 along our experiment. The `recovery` curve is our implementation and the `linear` is a simple additive function of current draw. The first strong decrease of the tenth minute is mainly due to traffic overhead and very few is from RPL control messages. Once the routing plane is established, between the tenth and twentieth minute, the battery decreases with a greater slope and with more micro variations. When removing nodes at the twentieth minute, the node 2 shows a quick and strong recovery of its battery lifetime. As given by the model, the fall of current draw lets the battery recovers some of its current charge. During the fiftheen following minutes the node sends one packet per second to the *Sink* and the curve is similar to the begining (without booting process) but at a lower battery level.

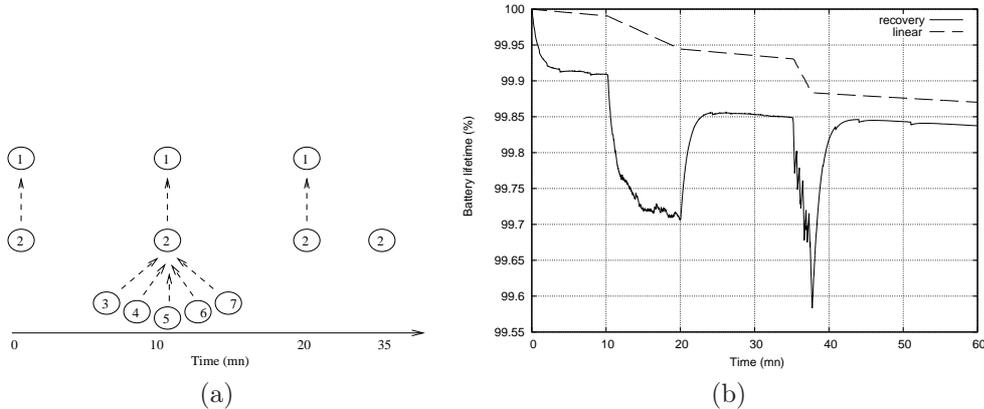
[1]that is only 0.0005% of the total

Figure 3: Battery lifetime and RPL

At the thirty fifth minute we remove the parent of the node 2. The very large decrease of battery is again mainly related to a strong use of communication. The node has no more IPv6 routeur neighbor then it polls the network with neighbour discovery messages and after three minutes, the node decides to build a new network and just send few RPL control message. One can there observe a very significant recovery at the part (b).

The linear model is very less reactive and never grows, as expected. This strong difference suggest it could over or under estimate the lifetime depending on how the other sensors behave.

4.3   Implementation Accuracy

The approximation we made on basic values and the rounding effect of divide operations lead to residual errors all along the lifetime estimation. We have implemented a floating point version of the model with a perl script and compare its result with the computation made by the mote. The figure 4 shows a snapshot of a WSN like those of the figure 3.a with the two battery lifetime estimations. The mote computes values a bit lower than floating point but the difference stays constant. Strong variations (between the minutes 30 and 35) have a much more amplitude but do not deviate the lifetime. These results enforce our confidence on our implementation as it follows the theory of battery consumption and especially the recovery one. We continue research on the simulation for long time duration.

4.4   Network life

Estimation of sensor lifetime is a prior information for their deployement. Sensors networks can be in place for several years before be replaced or recharged and so battery should be chosen to match the expected duration.

The Contiki operating system provides some Radio Duty Cycle (RDC) implementations we have used in order to compare their energy consumption. The part (a) of the figure 5 shows these experiments. All these RDC are asynchronous and packet oriented.

- the contikiMAC [3] allows the greatest battery lifetime. It can sleep up to 99% of the time and was measured as been ten times less energy consumer than the following X-MAC. When a node has to send, it transmits the packet several times, until the receiver return an acknowledge.
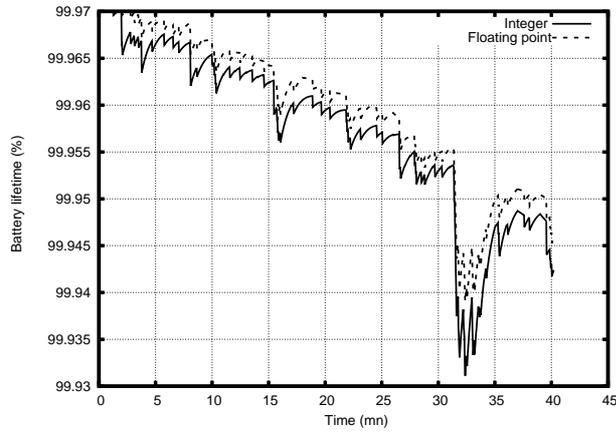
Figure 4: Floating point vs Integer battery lifetime computation

- X-MAC [1] is a more consumer; the sender sends several preambule packets until acknowledge reception and then sends the data packet.

- CX-MAC (Compatibility X-MAC) is a variation of X-MAC that is provided by Contiki to be usable on more radio chips.

- sicslowmac (also in Contiki) handles 802.15.4 frames and has its radio alway open. Note that with a TX at $\simeq 20mA$ and a battery at $880mAh$, the theory gives $\frac{880}{20} = 44h$ (cf table 3).
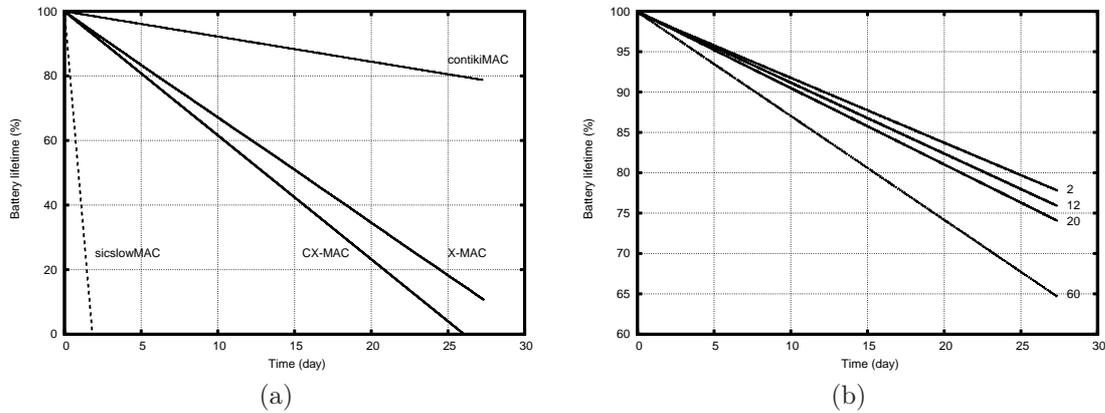

Figure 5: Network life

We have also measured the throughput effect on battery lifetime and plot them in part (b) of the figure 5. The contikiMAC RDC was used because it is the less battery consumer and so the impact of the throughputs is more discernible. Numbers within the part (b) are the throughput of the application, from 60 packets by minute ($pkts/mn$) to 1. Each packet has a size of 87 bytes and the application is parameterised by a number of seconds between two packets. The

figure shows that the lifetime is roughsly linear with the throughput but other tests with less throughput than $1pkt/mn$ have no significantly improved the lifetime. The network traffic is then negligeable against sensor base activity (indeed the routing protocol is the main radio user).

These figures are very close to a line (only a zoom shows their are not) and we use a linear regression with the least square estimation technique to envision the time when the battery will be completely empty (i.e. with 0% of remaining energy). The table 3 contains the battery lifetime estimation for each lines in figure 5. The short lifetimes of these networks (no more than four months) is related to the light battery we have used. Battery of node like Sky are usually around $2000mAh$ and should allow WSN be operational for one year. Nevertheless real batteries have a secure cut off level that prevent them to be fully discharged because they will be damaged and not able to be charged again.

RDC	Battery lifetime (days)	Throughput $(pkts/mn)$	Battery lifetime (days)
contikiMAC	128	60	77
X-MAC	30	20	105
CX-MAC	26	12	113
sicslowMAC	1.8	2	124
Throughput : $1pkt/mn$		RDC : contikiMAC	

Table 3: Battery life estimation

5   Conclusions

This paper presents a first implementation of a battery lifetime estimation inside an existing operating system for WSN. Based on a recognized theoritical battery model, our work has show how sensor internal architecture impacts on concrete realization. Our simulations help us to better understand how sensor use their battery in bootstrap process or during networking. We believe this work is usefull for many applications as routing optimization that is a work we currently plan. We also have to launch long time tests establishing a relation between the reamaining energy and the cut off level of a battery. Moreover several battery technologies have to be tested to enforce our implementation model.

References

[1] Michael Buettner, Gary V. Yee, Eric Anderson, and Richard Han. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, SenSys '06, pages 307–320, New York, NY, USA, 2006. ACM.

[2] C.B. des Roziers, G. Chelius, T. Ducrocq, E. Fleury, A. Fraboulet, A. Gallais, N. Mitton, T. Noel, E. Valentin, and J. Vandaele. Two demos using senslab: Very large scale open wsn testbed. In *International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, pages 1–2, june 2011.

[3] A. Dunkels. The contikimac radio duty cycling protocol. SICS Technical Report T2011:13 ISSN 1100-3154, Swedish Institute of Computer Science, December 2011.

[4] A. Dunkels, B. Gronvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *29th Annual IEEE International Conference on Local Computer Networks*, pages 455–462, November 2004.

[5] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He. Software-based on-line energy estimation for sensor nodes. In *Proceedings of the 4th workshop on Embedded networked sensors*, EmNets '07, pages 28–32, New York, NY, USA, 2007. ACM.

[6] T. Winter Ed. Rpl: Ipv6 routing protocol for low-power and lossy networks. RFC 6550, Internet Engineering Task Force, March 2012.

[7] F. Kerasiotis, A. Prayati, C. Antonopoulos, C. Koulamas, and G. Papadopoulos. Battery lifetime prediction model for a wsn platform. In *Fourth International Conference on Sensor Technologies and Applications (SENSORCOMM)*, pages 525 – 530. IEEE, july 2010.

[8] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level sensor network simulation with cooja. In *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, pages 641 – 648, nov. 2006.

[9] Debashis Panigrahi, Sujit Dey, Ramesh Rao, Kanishka Lahiri, Carla Chiasserini, and Anand Raghunathan. Battery life estimation of mobile embedded systems. In *Proceedings of the The 14th International Conference on VLSI Design (VLSID '01)*, VLSID '01, pages 57 – 63, Washington, DC, USA, 2001. IEEE Computer Society.

[10] J. Rahme and K. Al Agha. A recursive model for battery lifetime estimation in wireless sensor networks. Research report 1525, CNRS - Université Paris-Sud - LRI, September 2009.

[11] J. Rahmé and K. Al Agha. A state-based battery model for nodes' lifetime estimation in wireless sensor networks. In *Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '09, pages 337–338, New York, NY, USA, 2009. ACM.

[12] J. Rahme, N. Fourty, K. Al Agha, and A. Van den Bossche. A recursive battery model for nodes lifetime estimation in wireless sensor networks. In *Wireless Communications and Networking Conference (WCNC)*, pages 1 – 6. IEEE, April 2010.

[13] D. Rakhmatov and S. Vrudhula. An analytical high-level battery model for use in energy management of portable electronic systems. In *Computer Aided Design, 2001. ICCAD 2001. IEEE/ACM International Conference on*, pages 488 – 493, 2001.

[14] D. Rakhmatov and S. Vrudhula. Energy management for battery-powered embedded systems. *ACM Trans. Embed. Comput. Syst.*, 2(3):277–324, August 2003.

[15] D. Rakhmatov, S. Vrudhula, and D.A. Wallach. Battery lifetime prediction for energy-aware computing. In *Proceedings of the 2002 International Symposium on Low Power Electronics and Design (ISLPED'02)*, pages 154 – 159. ACM, August 2002.

[16] R. Rao, S. Vrudhula, and D.N. Rakhmatov. Battery modeling for energy aware system design. *Computer*, 36(12):77 – 87, dec. 2003.

[17] Paulo Sérgio Sausen, Marco Aurélio Spohn, and Angelo Perkusich. Broadcast routing in wireless sensor networks with dynamic power management and multi-coverage backbones. *Inf. Sci.*, 180(5):653–663, March 2010.

[18] H. Seungki, K. Daeyoung, and K. Jae-eon. Battery aware real time task scheduling in wireless sensor networks. In *11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 269 – 272. IEEE, August 2005.

[19] D. Timmermann. Simulation of mobile wireless networks with accurate modelling of non-linear battery effects. In *In proceedings of Applied simulation and modeling (ASM)*, 2003.

[20] P. Vasseur Ed., M. Kim Ed., K. Pister, N. Dejean, and D. Barthel. Routing metrics used for path calculation in low-power and lossy networks. RFC 6551, Internet Engineering Task Force, March 2012.

[21] Jiucai Zhang, Song Ci, H. Sharif, and M. Alahmad. An enhanced circuit-based model for single-cell battery. In *Applied Power Electronics Conference and Exposition (APEC), 2010 Twenty-Fifth Annual IEEE*, pages 672 – 675, feb. 2010.