Tuontu: A Tool for Evaluating the Impact of Wireless Sensor Network Design Alternatives

Barbara Staehle, Markus Leimbach, and Dirk Staehle

University of Würzburg, Institute of Computer Science, Würzburg, Germany
{bstaehle, markus.leimbach, dstaehle}@informatik.uni-wuerzburg.de

***Abstract*—With an increasing wireless sensor network (WSN) application complexity, more alternatives for the WSN design arose. This complexity is also the reason why it is difficult to asses how and at which price in terms of money or decreased quality of service, design alternatives increase the system performance. In this work we therefore introduce a concept for quickly answering likewise questions, namely the task-based resource consumption modeling. It is the heart of the framework Tuontu which allows to easily estimate if an application is feasible in a given WSN deployment and which performance is to expect.**

I. Introduction

For efficiently and successfully setting up or configuring a WSN, a potpourri of aspects have to be considered which are as different as hardware design choices, node deployment, protocol configurations, energy efficiency, costs, or end-user expectations. In 2004, Römer and Mattern [1] formalized this problem by describing a 14 dimensional WSN design space. Six years later, there are even more factors to consider as more sophisticated applications and hardware options appeared. Most of those factors important for setting up a WSN from scratch like e.g. sink placement [2] have been addressed by theoretical works. Many of those results are however not applicable for practical deployments, as not all used assumptions hold in reality. More helpful for configuring a productive WSN are manufacturer deployment guidelines [3] or experiences from WSN deployment campaigns [4].

A user which installs a WSN for environmental monitoring is often neither able to use an optimal combination of deployment strategies and protocols configurations, nor is she satisfied with general statements. Instead, she would simply like to know which advantages and disadvantages a certain design option has for her WSN application. Those insights need not be as detailed to require a lengthy simulation calibrated with hardware data [5], but should be more accurate than the ones provided by an Excel spreadsheet [6].

In this paper, we therefore pave the way towards a new concept for evaluating WSN design alternatives and introduce the *task-based resource consumption modeling* (TRCM) approach. The first pillar of this concept is to abstract the WSN to an amount of resources which are offered by the deployed nodes. The second pillar consists of decomposing each application into tasks whereof the resource consumptions are easily to determine. Those ideas are the core of a framework we call *Tuontu* as it allows to quickly analyze the impact of design alternatives and thereby helps users struggling with the question whether “To Use Or Not To Use” a certain feature.

This paper is structured as follows: In Section II we discuss related approaches. Details on TRCM and Tuontu are given in in Section III, and Section IV respectively. Section V contains numerical results illustrating the potential of our idea. We conclude and give an outlook on our next steps in Section VI.

This work was funded by the Federal Ministry of Education and Research of the Federal Republic of Germany (Förderkennzeichen 01 BK 0800, GLab). The authors alone are responsible for the content of the paper.

II. Related Work

A plethora of theoretical works on WSN optimization exist which propose thoroughly evaluated algorithms. However, environmental or hardware constraints are in general not included in those studies and therefore a challenge for a practical implementation. In the paper of Bogdanov et al. [2] for example, a base station positioning algorithm which optimizes the energy efficient operation of a WSN with energy harvesting nodes is introduced. As no restrictions on the base station locations are given, this algorithm is not applicable for an outdoor WSN deployment where base stations can only be deployed at locations where a power supply and a broadband Internet access are available. A network engineer can however still alter some protocol parameters or configure the interval at which the sensor nodes collect data. Consequently she would be interested in the trade-offs involved in this decision.

The size of the design space makes it impossible to deploy and test all possible configurations. A low-level simulation framework like the one presented by Hurni and Braun [5] could be adapted to the properties of the used hardware and used for a study revealing the impact of the degrees of freedom. The two major drawbacks of a likewise approach are however that the adaptation would be difficult and very likely not feasible by an end-user and require lengthy simulation studies in order to capture all interactions. At the other edge of the spectrum are application notes or helpful hints for a successful WSN deployment. Barrenetxea et al. [4] for instance share their experience from a number of WSN deployment campaigns. This allows to avoid obvious mistakes but does not help a person willingly to build up an own sensor network to rate the trade-offs of the specific decisions. This problems is partly attacked by the manufacturer Crossbow which provides guidelines for building a WSN based on application specific criteria [3]. This material gives e.g. advices on a suitable number of gateways for a given number of sensor nodes. Most users are however not satisfied with such general statements, as they have no change to rate, whether an additional gateway is worth the price of a faster data delivery and increased system lifetime without actually setting up a WSN.

Answering such questions is possible with Tuontu, as TRCM strives the balance between a quick but imprecise analysis and an accurate but lengthy and complex simulation. Note that Tuontu is no stand-alone optimization tool for a perfect WSN configuration, but does allow to quickly assess whether the use of a more detailed simulation of the impact of a certain design factor is worth the effort or not.

III. Task-Based Resource Consumption Modeling

In this section we discuss the two pillars of TRCM more closely. Firstly, a WSN is abstracted to an amount of resources, an idea we introduce in Section III-A. Secondly, each application running on top of the WSN is decomposed into tasks whereof the resource consumptions are easily to determine. In Section III-B we discuss this principle and its application for estimating the resource consumptions of an application and thereby to rate whether and how well it runs on a given WSN deployment. Please note that an exact analytical model is out of scope of this paper, in the following, we just use a formal language to sketch our main ideas.

A. Network Abstraction

For abstracting a WSN deployment $\mathcal{W}$, we define $\mathcal{N}$, with $N = |\mathcal{N}|$, to be the set of nodes in $\mathcal{W}$ which can be (Internet) gateways, relay or sensor nodes. The resources provided by node i are storage capacity, available energy and sensing capabilities, $z_i =< s_i, e_i, \pi_i >$. The *resource state* of $\mathcal{W}$ is hence defined as $Z_\mathcal{W} =< S_\mathcal{W}, E_\mathcal{W}, \pi_\mathcal{W} > \in \mathbb{R}^N \times \mathbb{R}^N \times \mathcal{P}^N$. The energy and storage resources are given by real numbers, whereas $\mathcal{P}$ denotes the set of perceptions which may be collected from the environment. π_i describes how node i perceives its environment, i.e. which characteristics of the physical environment the node can capture. For a typical sensor node, this could e.g. be $\pi_i = \{\texttt{nodestate}, \texttt{humidity}, \texttt{temperature}\}$. Gateway or relay nodes do not have sensors and can only report on their own operation condition, i.e. $\pi_i = \{\texttt{nodestate}\}$.

The *physical condition* $C_\mathcal{W} =< L_\mathcal{W}, O_\mathcal{W} > \in \mathbb{R}^{3N} \times \mathcal{O}^N$ of $\mathcal{W}$ gives the node locations and operation state. While $L_\mathcal{W}$ is constant for networks without mobile nodes, $O_\mathcal{W}$ is changing in accordance with the node activities. The possibilities for $o_i \in \mathcal{O}$ are given by the state machine used for abstracting the functionality of node i.

B. Applications and Tasks

As a typical WSN application α is rather complex, we decompose it to a set of tasks $\alpha = \{\tau_1, \tau_2, \ldots, \tau_n\}$. Tasks are basic functionalities accomplished by $\mathcal{W}$ and whereof the resource requirements can be simply determined. An example of a simple application α_0 is to let each sensor node report one temperature reading to the nearest base station. Hence, each node has to execute the tasks τ_m of measuring a temperature, τ_s to send it to its next hop and, if necessary, the task τ_f, to forward data. The upper half of Fig. 1 depicts the principle of application decomposition.

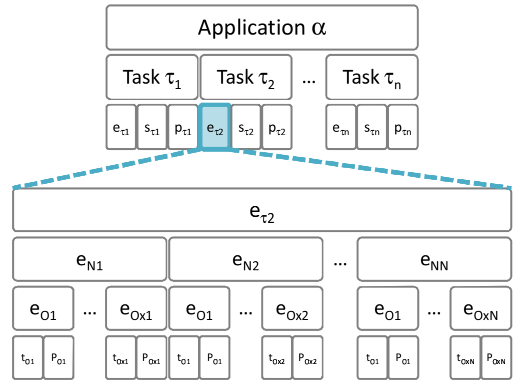

Fig. 1. Task-based energy consumption modeling

A task τ is characterized by its functionality and resource requirements. The functionality of τ_m for instance, is to measure and to store the temperature. The computation of its resource requirements, $Z_\mathcal{W}(\tau_m)$, and the mapping to the individual sensor nodes, in contrast, requires more details on the task functionality and on the network configuration. This is done by the mapping function m:

$$m : (\tau, \mathcal{W}) \mapsto Z_\mathcal{W}(\tau) =< S_\tau, E_\tau, \pi_\tau > . \quad (1)$$

In the lower half of Fig. 1 it is exemplarily depicted how the energy consumption, $E_\mathcal{W}(\tau_2)$ of task τ_2 are computed as the joint node energy consumptions $e_i(\tau_2)$, where $1 \leq i \leq N$. Likewise are $S_\mathcal{W}(\tau)$ and $\pi_\mathcal{W}(\tau)$ composed of the individual node storage consumptions and perception requirements. $e_i(\tau_2)$ in turn, is the addition of the energy consumptions ε_{o_j} of each of the operational states $1 \leq j \leq x_i \in \mathcal{O}$ node i is in during the task execution. Finally, the operational state energy consumptions ε_{o_j} are computed as the product of state power consumptions P_{o_j} and the time t_{o_j}, node i spends in state o_j.

Note that the implementation of m has to capture the different resource consumptions on different types of nodes. Additionally, it is possible to model applications where not all nodes do the same thing, as only some nodes need to measure and or process data while others do nothing except forwarding data if necessary. In the special case where a task is not feasible on $\mathcal{W}$, e.g. if the collection of sensor data from an area where no sensor nodes are available would be required, it requires no resources on $\mathcal{W}$, i.e. $m(\tau, \mathcal{W}) \ =< 0, 0, \emptyset >^N$.

The effect of an application α on $\mathcal{W}$ is determined by the *application execution function*

$$X : (\alpha, Z_\mathcal{W}) \mapsto \chi(\tau_n, \chi(\tau_{n-1}, ..., (\chi(\tau_1, Z_\mathcal{W})))). \quad (2)$$

which is iteratively computed by the *task execution function*

$$\chi : (\tau, Z_\mathcal{W}) \mapsto Z'_\mathcal{W} = Z_\mathcal{W} - m(\tau, \mathcal{W}) = Z_\mathcal{W} - Z_\mathcal{W}(\tau). \quad (3)$$

$\chi(\tau, Z_{\mathcal{W}})$ in turn is for $1 \leq i \leq N$ defined as

$$z_i - z_i(\tau) \ = < e_i - e_i(\tau), s_i - s_i(\tau), \pi_i - \pi_i(\tau) > . \quad (4)$$

The "sensing capability" resource does not need to be reduced, as sensors are not changed by the execution of tasks. Instead, we define for $p, q \in \mathcal{P}$

$$p - q = \begin{cases} p & \text{if } q \subseteq p \\ \emptyset & \text{otherwise.} \end{cases} \quad (5)$$

Consequently, χ yields a resource state, where at least one sensor node has an empty perception set if a task requires the collection of a measurement area where no adequately equipped sensor nodes are existing. Together with $m(\tau, \mathcal{W})$ yielding a zero resource consumption if the task is not feasible, a simple sanity check if during the computation of $X(\alpha, Z_{\mathcal{W}})$ a not changed resource state or a state with negative or empty entries is reached is hence sufficient for checking whether α is feasible on $\mathcal{W}$ or not.

IV. TUONTU

Tuontu uses the previously introduced concepts, in order to determine, if and with which performance an application may successfully be executed on a given WSN deployment. To model imperfectness of hardware and the harshness of the environment, we additionally include random factors in the computation of the resource consumptions. Tuontu is implemented in Java and intentionally kept modular to make it easy to include more WSN design factors than the ones reviewed in Section IV-A we used for our initial experiments. The performance metrics which are currently implemented are described in Section IV-B.

A. Factors under Consideration

In this section we walk through the plethora of factors characterizing a WSN deployment considered by Tuontu in a bottom-up fashion. The *deployment area* characterizes the size of the area to monitor and optionally candidate locations for the node positions. The *deployment strategy* gives the location of the nodes. As an abstraction of more or less sophisticated placement algorithms, we consider the sensor nodes to be either deployed on a regular grid, in a random fashion or clustered according to the importance of the area to monitor. Another factor influencing the physical network condition C, the task resource requirements and thereby the functions m and χ is *the used hardware*. As a starter, we use the Crossbow eKo node [7] as a role model to derive the node state machine and the corresponding power and time consumptions. We also adopt eKo characteristics for properties like size of the RAM and flash memory, sensing capabilities and the networking stack. Furthermore are the possibilities for the *energy supply* inspired by the eKo capabilities. We assume the most common setup where the sensor and relay nodes in the field have 2 AA batteries, whereas the gateways are mains powered. Additionally, the user could decide to augment the nodes with an energy harvesting unit or to go out and exchange batteries if necessary. The amount of energy gathered by the solar energy harvesting process is modeled to be normally distributed over the daily sunshine duration and parameterized according to [9] and [7].

In general, the *networking stack* depends on the used hardware. A packet-level simulation is not our goal, we therefore use an abstract networking stack. It is based on the IEEE 802.15.4 [8] physical layer channel model extended by a shadow fading component parametrized to result in an average link length slightly larger than 250 m [7]. As a low-power MAC protocol, we abstract a solution similar to CSL proposed by the upcoming 802.15.4e [10]. CSL enables low-power multi-hop communication at the price of an increased delay by periodic channel scans each σ seconds which is also the length of the preamble proceeding each packet. The routing topology is abstracted to a minimum hop topology which requires a certain amount of energy to be constructed and to be self-healing. On application layer, we abstract the possible *applications* to do either periodic data reporting, to report the occurrence of random events or to answer user inquiries. The *data sampling period* δ gives the length of the interval between two periodical activations of the sensors. The *application intelligence* determines to which degree the nodes do process the data. At the moment we namely consider the effects of a simple data aggregation protocol, where each forwarding node has to wait for the data packets of its children in order to forward them together with its own data.

B. Performance Metrics

The *network lifetime* is clearly the most important metric for a WSN design. As countless "lifetime" definitions exist, we use the time when 50% of all nodes have run runs out of energy as "lifetime" which is in any case an indicator for the network longetivity regardless if the WSN is still functional after this period or not. The *application layer performance* is characterized by the quality of data and the data delay. For this study, we use the average data delivery delay as application layer performance metric. It is estimated as the product of the path length and the sum of the preamble length σ and the packet transmission time and additionally includes the effect that packets might have to wait for being aggregated. Note that the topology and thereby the data delivery delay changes with the time, we therefore use the initial path length for this computation. The third, metric are simply the monetary *costs* of the deployment.

V. RESULTS FROM A FACTORIAL DESIGN STUDY

The goal of Tuontu is to give insights how a given WSN deployment can be optimized. For this purpose, we distinguish between *hard factors* which can not be influenced and *soft factors* which can be adapted. In this section we illustrate the potential of our idea by reporting on the results of a factorial experiment which visualizes to what degree and at what price soft factors may influence the system lifetime.

A. Experimental Setup

We assess the influence of the soft factors wherefore we show exemplary "high" and "low" values in Table I by repeating the same experiment for each of the resulting 2^5 design points in 50 different WSN topologies. One network snapshot consists of 50 sensor nodes randomly spread in a 500×500 m square with the gateway(s) at its corner(s). All factors not mentioned in Table I are considered to be "hard" and parametrized as discussed in Section IV-A. The input files for Tuontu summarize this setup and are publicly available[1].

TABLE I
LOW (-) AND HIGH (+) SOFT FACTOR LEVELS

soft factor	level (-)	level (+)
number of gateways G	1	4
energy source	battery	solar
data aggregation	off	on
sampling interval δ	5 min	15 min
channel scan period σ	5 s	20 s

One experiment consists of running an application mix of regular data reporting, random event detection and answering user queries on top of the WSN. The experiment ends either after 2 years, or when 50% of all nodes are out of energy.

B. Numerical Results

At the end of each experiments, the metrics system lifetime, average packet delay and cost are collected. The influence of factor x on the system performance in terms of metric y is characterized by its *main effect* $e_x(y) = (\bar{y}_{x^+} - \bar{y}_{x^-})/2$, where $\bar{y}_{x^+}$ and $\bar{y}_{x^-}$ denote y averaged over all design points where x is at its high level and low level respectively. $e_x(y)$ hence simply expresses which average impact setting x from its low value to its high value has, regardless all other factors. Fig. 2 visualizes the main effects of soft factors on the system metrics. The 95%-confidence intervals demonstrate, that most of the effects are statistically significant.

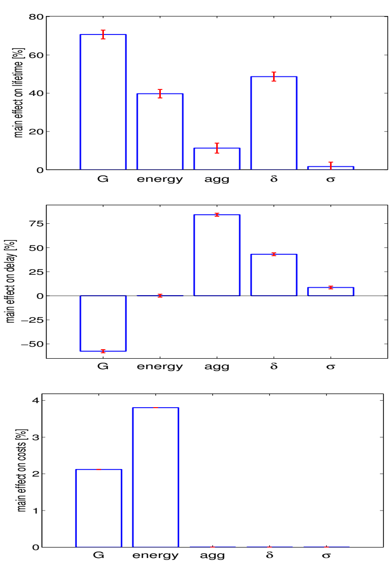

Fig. 2. Main Effects of Soft Factors

[1]http://www3.informatik.uni-wuerzburg.de/staff/bstaehle/tuontu/

As the absolute main effects depend strongly on the WSN configuration, we show normalized main effects only in order to allow a better comparison. For actually deciding about a certain WSN configuration, the absolute values are of course necessary. This form of representation is however suitable for pointing out promising optimization direction. Fig. 2 hence confirms the intuitive assumption that choosing for all soft factors the high instead of the low value always increases the system lifetime. It also shows that some factors have a stronger influence than others and that the negative effect of some factors are different. More precisely has a larger number of gateways the strongest impact on the system lifetime, as this would lead to shorter paths, thereby reducing not only the forwarding load of the sensors, but also the average packet delivery delay. This is hence a promising way for increasing the system lifetime and performance, if the user is willingly to pay the price for the hardware. As we assumed the price for 50 solar panels to larger than the one for three additional gateways, using energy harvesting instead of batteries would in this case be the more expensive option, but this solely depends on the chosen numbers. Note however also that cost neutral options, like the use of data aggregation, longer sampling periods, or longer channel scans are also suitable for increasing the system lifetime. A closer analysis of the interactions of different parameters and implementations for those combined factors could hence be an interesting optimization option.

VI. CONCLUSION AND OUTLOOK

This work introduced the idea of task-based energy consumption modeling for wireless sensor networks. It is the heart of Tuontu, a tool for evaluating the impact of WSN design decisions on the system lifetime and performance. Results from an exemplary factorial design study demonstrate the soundness and applicability of our idea. The refinement of Tuontu is ongoing and includes the integration of more factors and metrics for allowing a holistic WSN optimization and extensive parameter studies.

ACKNOWLEDGMENTS

The authors would like to thank Clemens Mühlberger and Marcel Baunach for help on technical questions and Phuoc Tran-Gia for his support and valuable comments.

REFERENCES

[1] K. Römer and F. Mattern, "The Design Space of Wireless Sensor Networks," *IEEE Wireless Communications,*, vol. 11, no. 6, 2004.
[2] A. Bogdanov *et al.*, "Power-aware Base Station Positioning for Sensor Networks," in *INFOCOM*, HongKong, China, 2004.
[3] G. Baleri, "Guidelines for WSN Design and Deployment," Crossbow Technology, Inc., 2008.
[4] G. Barrenetxea *et al.*, "The Hitchhiker's Guide to Successful Wireless Sensor Network Deployments," in *SenSys*, Raleigh, USA, 2008.
[5] P. Hurni and T. Braun, "Calibrating Wireless Sensor Network Simulation Models with Real-World Experiments," in *Networking*, Aachen, Germany, 2009.
[6] Crossbow Technology, Inc., http://www.xbow.com.
[7] Crossbow Technology, Inc., "eKo Pro Series, eKo Components - For Environmental Monitoring," Data sheet, 2009.
[8] IEEE Computer Society, "IEEE 802.15.4 Standard for Information technology," 2006.
[9] J. Taneja *et al.*, "Design, Modeling, and Capacity Planning for Micro-Solar Power Sensor Networks," in *IPSN*, St. Louis, USA, 2008.
[10] IEEE Computer Society, "IEEE P802.15.4e/D0.01 Draft Standard for Information technology," 2009.