Inferring Technical Constraints of a Wireless Sensor Network Application from End-User Requirements

Felix Jonathan Oppermann
Department of Computer Science
Carl von Ossietzky University of Oldenburg
26111 Oldenburg, Germany
Email: oppermann@informatik.uni-oldenburg.de

Steffen Peter
IHP GmbH
Im Technologiepark 25
15236 Frankfurt (Oder), Germany
Email: peter@ihp-microelectronics.com

***Abstract*—This paper describes a two-step process to infer specific technical constraints and parameters needed for a reliable mission-specific design of wireless sensor networks (WSN). As the first step, we propose a new requirement catalog helping end-users to formulate a complete and consistent specification of WSN mission requirements. Its generality allows the unambiguous characterization of a wide spectrum of applications from the end-user's perspective. As the second step, we introduce a methodology to deduce fine grained technical specifications from the general requirements. The proposed automatic graph-based requirement expansion approach translates the content of the catalog and additional requirements to specific technical terms, which provide the basis for an application-specific WSN design. A real-world use case – a new WSN application in the area of critical infrastructure protection – demonstrates the applicability of the presented approach.**

***Keywords*-Requirements/Specifications; Design Tools and Techniques; Sensor Networks;**

I. Introduction

As of today, advantages in microelectronics permit the equipment of individual sensors with limited computing capabilities and radio interfaces. It is envisioned to apply such wireless sensor networks (WSN) in a wide range of different scenarios, including application areas like habitat and structure monitoring, catastrophe management, and home automation. The diversity of the application range and the specific properties of WSNs make their design difficult and especially challenging for the intended end-users, like biologists, geologists, and engineers. To allow a widespread use of WSNs, it is necessary to enable technically less skilled people to successfully deploy a WSN. While these end-users can be assumed to be experts in their application domain, for example in biology or in geology, they cannot be expected to be especially trained in computer science in general or WSN technology in particular. One must ensure that the design process requires as little technical knowledge as possible and can be automated as far as possible.

Often, a large number of alternative solutions is available and it is difficult to select a good one. Naturally such a selection needs a clear definition of what is actually required. Thus, the definition of requirements is one key issue during the development of WSNs. Only with precise information of what should be achieved it is possible to perform precise and goal oriented engineering. To the best of our knowledge, currently no accepted and reliable way of formally specifying WSN applications and their parameters is known. A major problem is the diversity of application domains. Another problem is that domain experts are usually not familiar with the terms used in WSN engineering. Thus, a translation of rather high level, fuzzy, domain-specific requirements to measurable metrics that can be used within the WSN development process is required.

This paper proposes a two step approach to infer the required precise technical parameters from the general end-user requirements. After the description of the WSN design flow in Section II, the requirement definition process is introduced. The first step of the requirement process applies a novel requirement catalog helping end-users to formulate complete and consistent specifications of WSN applications. This requirement catalog is described in Section III. Section IV describes the second step, a graph-based requirement expansion process which outputs a fine granular technical specification. A real-world use case, a new WSN application in the area of critical infrastructure protection, demonstrates the applicability of the proposed approach in Section V. Finally, the paper is concluded by a summary and brief outlook in Section VI.

II. Design Process

It is our vision that in the future the design of WSN application is not the expensive and extremely time consuming development task it is today, but a rather straightforward process. Ultimately, it should even be possible for end-users to execute this process on their own.

Two distinct approaches have been proposed towards a simplified design and setup of WSNs. The first approach is based on a standard middleware that is deployed on all nodes. It is configured according to the requirements of the application. TASK [1] is a prominent examples for such configurable middlewares in sensor networks. The approach

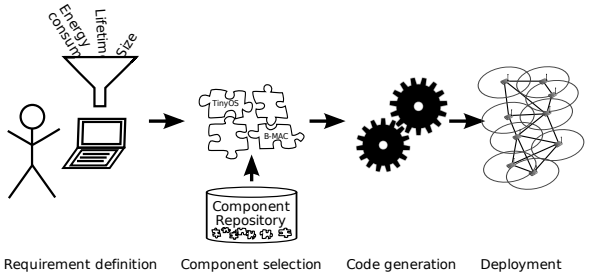

Figure 1. Stages of our envisioned partly automated WSN design process

is promising and appears to be feasible for application scenarios that are not too diverse and if sufficient memory and processing resources are available. The second approach includes a (semi-)automatic composition process. Examples are SNACK [2] and ConfigKIT [3]. Based on the application requirements, exactly the components (i.e., hardware and software modules) that promise to deliver the required task optimally are assembled. Hence, it promises to be highly efficient with regard to memory and computational costs. A major disadvantage of such a composition process is its technical complexity: how to find modules that can be composed and how to predict how such a composition behaves? In this paper, we focus on the latter approach, since we are convinced that coping with stringent memory and computation constraints – a key benefit of the composition process – will stay a key factor in WSN engineering. In particular, we pick up the development process introduced by ConfigKIT [3]. ConfigKIT allows the automatic selection of suitable components from a repository based on abstract requirements. In contrast to the broader scope of the approach presented in this paper, the focus of ConfigKIT is primarily on security aspects.

Figure 1 shows the general flow of a composition driven design process. Based on the requirements defined by the user, components and software modules are selected that promise to satisfy the user's requirements. The selection and assessment process employs a repository containing models of components and their properties. Based on the resulting configuration, the actual source code is generated and compiled. Finally, the sensor nodes are equipped with the resulting code images and are deployed at the application site.

It is apparent that the basic requirements given by the user define and start the process that finally concludes in the deployment of the sensor network. They define the constraints, the environment, the functional and qualitative properties that have to be met. Thus, they influence each following step in the development process. The requirement definition step can only be done in close co-operation with the future users of the WSN. To allow such a co-operation, it is necessary to agree on a common language for this phase. On the one hand, this language needs to be understandable for the users and on the other hand it needs to allow a complete specification of all important aspects of the mission.

From our experience, in most cases such a requirement definition – even if it is correct, consistent, and complete – cannot be used directly to start the composition process. Rather, it is necessary to translate the user requirements into a technical specification, containing terms such as "topology", "data rate" or "message integrity". In ConfigKIT, the responsibility for this translation was mainly shifted to the user side. It is expected that the user is able to understand and to define the technical terms. Thus, this tool is clearly focused on developers and not on less experienced end-users. In order to support end-users directly, a more general requirement formulation step has to be added before defining the actual technical WSN terms. We presume that end-users are able to formulate abstract application requirements sufficiently and correctly. However, since we do not assume they can use technical WSN terms correctly, a deduction step is required to infer the technical terms from the requirements given by the user.

This process poses two general questions. First, how can a user enter the application requirements so that they are understandable to him and the WSN engineer, and also usable in a formal framework? And second, how to deduce the technical terms from the (partially fuzzy) end-user requirements? To tackle the first question, we propose a catalog of possible requirements that can structure the requirement definition process and allows a more formal specification of the application. In addition, it may help to ask the right questions when communicating with the future users. A detailed description of the catalog can be found in the next section. In a second step, as described in Section IV, the resulting requirement definition is translated to a more detailed technical specification.

III. Requirement Catalog

In this section we present a catalog of WSN requirement dimensions that is intended to structure the requirement analysis for WSNs. Even though different in aim and scope, this catalog has some similarities with previously proposed WSN classification schemes. In 2002, Tilak, Abu-Ghazaleh and Heinzelman [4] defined an early taxonomy for WSNs. This taxonomy allows a classification of WSNs according to different communication functions, data delivery models, and network dynamics. Römer and Mattern [5] define a list of properties that allow the characterization of WSN applications. Their design space contains 12 major dimensions, some of which contain several sub-dimensions. Similar to the taxonomy defined by Tilak et al., their focus is on communication and topology aspects. This classification scheme was later refined by Rocha and Gonçalves [6]. The result is a simplified classification scheme with only seven major dimensions. All of these classification schemes have

in common that they do not only consider requirements and constraints, but also more technical properties of an actual deployment, like network size, network topology, and heterogeneity. These properties are useful for classifying existing WSN solutions, but are unsuited for structuring the requirement analysis.

The requirement dimensions of the proposed catalog can be assigned to five categories. A complete overview is given in Table I. Each dimension is either specified by possible instances (italics) or a short definition. In the following sections, we examine the dimensions in more detail.

A. Mission

This category groups the functional requirements of the application, which define the goal of the WSN deployment. A central factor is the selection of suitable *sensors*, which determine what can be detected and monitored by the sensor network. Furthermore, this already defines several properties of the WSN. In addition to a selection of sensors we also specify how often the sensors need to be read. The *sampling interval*, combined with the properties of the sensor, defines how much data is generated over time or, in case of event detection, how often the node needs to evaluate whether an event has been detected. The sensory range of typical sensors usually found in WSNs is often rather restricted. Consequently, full *coverage* of a given area requires a large number of nodes, but many application scenarios require only partial coverage of the area and thus a lower number of sensor nodes. If the exact *number of measurement* points is known to the user, it is helpful to allow him to directly specify this number. A sensible interpretation of the data gathered by a WSN often requires a *spatial* and *temporal correlation* of the individual measurements to generate a coherent picture of the situation. In particular, some scientific applications, like the monitoring of earthquake shock waves, can require very precise information of the time and location at which an event is detected [7]. The demanded temporal and spatial accuracy determines the need for time synchronization and localization and the degree of precision these algorithms have to provide. The *mode of operation* defines how the sensed data is to be retrieved by the user. Scenarios range from *event detection*, where the user is only alerted if some predefined event is detected, to the *monitoring* of a given area up to more *interactive* systems replying to the user's queries on demand. The mode of operation implies how much intelligence is required inside the WSN. The degree of *mobility* in the network has a strong influence on routing and media access. The choice of localization methods is also influenced by the degree of mobility.

B. Operation Environment and Deployment

Besides functional requirements of the mission, the design of WSNs is also largely affected by the properties of the operation environment. The most important environmental aspects are probably the *size* and *dimensionality* of the deployment space. The size of current WSN deployments is quite diverse and ranges from few nodes in a single room to thousands of nodes spread over several square kilometers. Combined with the desired coverage and the network dynamics, the size determines the number of nodes that are needed for the deployment. Often overlooked is the dimensionality of the deployment space, although it has a strong impact on routing algorithms and positioning systems, as many common algorithms only work well in a two-dimensional network. We consider the space to be two-dimensional if one or two dimensions dominate in size and no nodes need to be placed above each other. The exact *environmental conditions* also largely influence the design. As a starting point – analogously to Römer and Mattern [5] – we only differentiate between outdoor, indoor, and mixed environments. A fully automated design process, especially if also considering hardware design, might require a more detailed distinction. As an exception, we included an additional more detailed description of the constraints for radio communication as these have a strong impact on the design of a WSN. WSNs are usually envisioned to operate autonomously in remote locations, but in many scenarios, this assumption does not hold. In case it is known that some infrastructure is available or the WSN is well accessible, it makes sense to exploit this. How the WSN is going to be deployed at the final operation site can also significantly affect the design of the sensor network. A careful manual placement of all nodes puts less demand on self-organizing qualities of the routing method than randomly dropping the nodes from an unmanned aerial vehicle (UAV). In some scenarios it is possible or desired to modify the network during its use and, for example, to add further nodes in order to replace failed ones or extend the monitored area.

C. Performance and Dependability

To be a useful tool, a WSN needs to live up to performance and dependability expectations. While early environmental monitoring applications seldom had challenging performance requirements, this changed with recent safety critical applications, like forest fire detection. As sensor nodes are usually powered by non-replenishable energy sources and WSNs are intended for extended operation periods, lifetime is probably the most important non-functional property of a WSN. There are several definitions of WSN lifetime. In this context we define *lifetime* as the amount of time the network can operate until too many nodes fail and other requirements are irrevocably violated. Expected lifetime may range from hours to several years. A high lifetime goal puts strict limitations on other properties of the WSN. In some scenarios, energy harvesting could significantly increase lifetime, but in current applications it is rarely used. In general, it is expected that a WSN is always operational during its lifetime, even in the presence of failures. For a

Table I
REQUIREMENT CATALOG

Category	Requirement Dimensions	*Instances* / Definitions
Mission	sensors	list of sensors
	sampling interval	minimal interval to sample the sensors
	coverage	*points of interest, sparse, dense, redundant*
	number of measurement points	number of required sensors (if known)
	accuracy of spatial correlation	maximal position error for measurements or events
	accuracy of temporal correlation	maximal error of measurement/event timestamps
	mode of operation	*event detection, monitoring, tracking, interactive*
	mobility of sensors	*static, partly mobile, mobile*
	mobility of observer	*static, mobile*
Operation Environment and Deployment	dimensionality	*two-dimensional, three-dimensional*
	size	maximal dimensions of the deployment space
	environment conditions	*indoors, mixed, outdoors*
	radio interference level	*none, low, medium, high*
	radio regulations	country/region of deployment
	available infrastructure	list of infrastructure systems (e.g., GPS, power grid)
	mode of deployment	*manual, random*
	time-frame of deployment	*one-time, continuous*
	accessibility	*inaccessible, limited accessibility, accessible*
Performance and Dependability	lifetime	minimal time until the WSN permanently fails
	availability	percentage of lifetime the network is operational
	channel dependability	percentage of reported events out of all locally detected events
	response time	maximal time between event detection and report
Security	eavesdropping resistance	*none, low, medium, high*
	tampering resistance	*none, low, medium, high*
	denial of service resistance	*none, low, medium, high*
	access control	*none, monitored, authorized, restricted*
	stealthiness	*none, limited*
Development Costs	monetary costs	maximal overall costs
	development effort	maximal man-hours

number of scenarios, it is possible to trade availability for higher lifetime or reduced costs. We define *availability* as the percentage of the lifetime the network is operational and can respond to queries. In addition the user's expectations on the *dependability* and *response time* of the communication need to be defined. There is usually a conflict between certain dependability and latency constraints and lifetime goals. A low latency, for example, reduces lifetime as it prevents long sleep periods for the nodes.

D. Security

For early WSN applications, security was not a concern. Typically, WSNs were deployed for habitat monitoring or similar scientific applications. Data secrecy is usually no concern in such applications and tampering is unlikely especially if the WSN is deployed in a remote location. Other WSN applications pose demanding requirements in terms of security. For example in medical applications privacy of the sensed data needs to be protected and obviously military or security applications demand for reliable operation even in the presence of attacks. We differentiate four orthogonal security dimensions. If the data generated by a WSN is of privacy critical nature, it is necessary to prevent easy *eavesdropping*. Eavesdropping can be countered by using protected communication channels, either by using physically secured channels or by applying cryptographic means. If the soundness of the reported data is important and it is likely that an attacker tries to manipulate the communication, additional protection is necessary. *Tampering* can be countered by employing authentication. Finally, an attacker could also try to completely interrupt the operation of the network. For all the above dimensions, the required level of protection depends on the capabilities of a likely attacker. We distinguish four security levels by applying an attacker classification, similar to the classification scheme proposed by Abraham et al. [8]: no protection, individuals accidentally detecting and playing around with an unprotected network, small groups with limited resources and knowledge, and large organized and experienced groups that carry out planed attacks. The required level of protection also depends on what kind of access control is already enforced in the operation environment. If the nodes are physically well protected, only attacks on the radio channel are likely. The classification of *access control* measures is based on a similar scheme by Weingart et al. [9]. The last security dimension describes the *stealthiness* of the WSN. Especially in military and security applications, it can be required to conceal the presence of the network.

E. Development Costs

Besides technical aspects, available funding and development capacities can play an important role for design decisions. Especially for large scale networks, it is important to limit *monetary costs* of individual nodes and limited

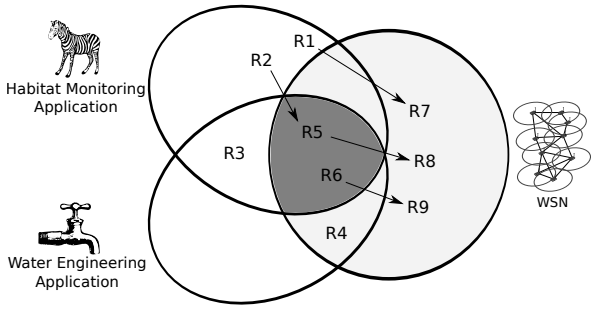

Figure 2. Requirement design spaces: Each ellipse represents the space of requirement types that are understood by corresponding domain experts. It is the goal to infer requirements understood by WSN engineers (R4 to R9) from domain-specific requirements (R1 to R3). Key is a requirement catalog (dark gray area, contains R5, R6) understood by all experts.

development resources might, for example, disallow the excessive use of custom built components.

IV. Requirement Expansion

It is the goal that the requirement catalog introduced in the previous section covers the space of possible requirement combinations as broadly as possible. Nevertheless, the space will never be complete in a way that it can describe all possible application requirements. This is mainly caused by the notion that the catalog space only covers requirements that are well understood by experts of all domains. Naturally, there remain requirements that do not fit in such a general catalog. Additionally, new requirements can occur for new application areas. In Figure 2 the potential catalog area is the shaded shared space in the middle, that only overlaps a fraction of the likely requirements. Each ellipse in Figure 2 represents the space of requirement types that are understood by corresponding domain experts, for example habitat monitoring designers or waterworks operators, on the left and the domain of WSNs on the right. Overlapping areas represent shared knowledge space. Basically, there are two means to cope with the problem of undefined requirements. Either the catalog is extended by new dimensions, or additional specifications beside the actual catalog are allowed. Extending the catalog to less general categories and properties is not appealing. It would render the catalog cluttered and thus less usable, and contradict the initial notion that the catalog is a shared space for all domain experts. Therefore, it is inevitable that requirements exist outside of this catalog on both sides. On the one side, the end-user has requests that are not cataloged and are not initially understood by WSN engineers. For instance in ZebraNet [10] the nodes are deployed to zebras. A fact from which a biologist can easily imply other requirements based on the behavior of zebras as herd animals, while that knowledge is not common for computer scientists. On the other side, WSN engineers eventually need technical definitions that are beyond what end-users have to know, for example the need for and parameters of congestion protocols in the transport layer.

Figure 2 illustrates the three different types of possible deductions:

- inferring requirements into the catalog (R2 $\rightarrow$ R5),
- inferring requirements from the catalog to core technical definitions (R6 $\rightarrow$ R9),
- inferring requirements alongside the catalog (R1 $\rightarrow$ R7).

The first type has already been tackled by the catalog itself, since its task is to raise the questions that support users entering correct requirements. The latter two inferring types can be resolved by a forwarding process, we call "requirement expansion."

The requirement expansion process works on a flexible graph structure containing all known requirement types. Requirement types are types that can (but do not need to) be set for applications under development. The 29 dimensions in the requirement catalog already define 29 different requirement types. The basic idea of the graph is that once certain requirement types are defined, they implicitly also defines other deducible requirement types. For example, from the fact that zebras are herd animals we can deduce that many similar nodes are in range, so that the density of the network is rather high, which can be inferred to multi-hop, short distance communication requirements.

We apply this forwarding methodology for deducing requirement types of high abstraction to technical terms, but also within the technical requirement space. The latter is motivated by the fact that the same requirement can be expressed in different ways. For example, the sampling frequency can be expressed as sampling period. It is our goal that all possible expressions of a requirement are set, especially with regard to an automatic composition process.

The underlying graph structure G can be expressed as directed graph $G = (R, T)$, while R is the set of requirements types, and T are the edges describing the translation of derivable requirements. Each element R_i out of R is a tuple $R_i = (D, V)$, D is the name or description of the requirement dimension; and V is the description of the value space. The value space can define a numeric range or a nominal scale, as needed for most of the requirements of the catalog. The translation set T is the set of triples $T_j = (R_{\text{from}}, R_{\text{to}}, f)$, which describe the mapping f from requirement R_{from} to R_{to}. The mapping function f can be an arbitrary function. In most cases basic math operations and conditional expressions are sufficient.

Without changing this general semantics, in practice we found it valuable to include the translations in the description of individual requirements R_i. By this, individual requirements contain information on how they relate to other requirements. To realize this, we followed an optional push/pull methodology. *Push* means that a requirement translates its properties to neighbored requirements. As an example, the translation from qualitative requirement metrics

to absolute values is usually a push from the qualitative properties. This allows the existence of more than one qualitative metric in the system to describe the same property. In the opposite direction, a requirement type can *pull* parameters of other requirements to define its own settings. For example, a data throughput requirement can pull values of packet size and measurement interval to infer its setting.

Adding the push and pull translation R_i can be refined by the tuple $R_i = (D, V, T_{\text{push}}, T_{\text{pull}})$. The set T_{push} is a subset of T, where $R_{\text{from}} = R_i$. The set T_{pull} is a subset of T, where $R_{\text{to}} = R_i$. This allows to build G solely on the information stored in the requirements.

Since as a result of this structure translations are part of the elements, new requirement types can be easily added without interfering with existing structures. For instance, if a domain engineer wants to add a requirement without using the catalog (like R1 in Figure 2) , it is sufficient to describe D and V of the new type, and add push translations to describe how the new requirement affects already existing requirement types. In the habitat monitoring example, R1 could be the species of monitored animals, and the translations describe how the behavior of the animals affects network properties. In contrast, novel technical terms typically apply pull translations. For example a new metric for expressing energy consumption would describe how it can be deduced from existing types, that is how it is connected to the existing knowledge. The modular approach of describing requirements allows adding requirement types in a flexible way. It is even possible to bundle requirement types in packages that can be added to the core database (catalog and WSN types) based on the application domain under development.

V. Case Study: water pipe surveillance

In the following, we demonstrate the applicability of the proposed methodology by using it to conduct the requirement analysis for a recent WSN application. The scenario is part of the WSAN4CIP project [11]. The *Frankfurter Wasser- und Abwassergesellschaft mbH* (FWA) demonstrator concerns the fail-safe and secure data transmission for monitoring the operation of water mains at Frankfurt (Oder), Germany, and thus to protect that type of critical infrastructure. It is mandatory that the system can provide a similar degree of reliability and security as wired monitoring systems, which are currently used for the task. By reducing the necessary infrastructure, a WSN should allow to reduce costs and provide greater flexibility.

The pipeline connects the waterworks, where the drinking water is collected, and an elevated tank by two parallel underground water pipes with a total length of 17.5 km. Part of the infrastructure are five stations along the pipes that are equipped with flow rate and pressure measuring devices permitting optimal operating and monitoring of the pipe system. The gathered data is displayed at the central process management system for supervisory control. The state of the system is measured every 30 seconds. The projected WSN consists of five measurement delivering nodes that are placed along the pipe at a distance of up to 5 km. The nodes will be located at the existing monitoring stations that provide information on water pressure and flow rate as analog data. Additional nodes are placed between the measuring nodes to relay the network traffic.

In the manual process, as first step, the WSN engineers, probably in dialogue with the end-user, has to derive the technical specifications. If, in contrast, the proposed requirement definition catalog is applied, the clearly structured initial requirements would look similar to Table II. The table lists the determined requirement values for the FWA demonstration scenario. These 29 requirements are the output of the user definition phase. Following the translation flow as described in the previous section we could derive many technical requirements. For example, we directly derive the properties of the sensors. These derived properties (floating point values as data format and a measurement time of less then one second) are new constraints for the further development process. Thus the sensor types *pushed* the new requirements as introduced in the previous section. The expanded requirement space directly feeds the selection process as introduced in Section II.

While the catalog and the requirement expansion in this example could demonstrate the general suitability, also several problems were discovered: (1) The translation from the catalog to the technical terms still needs human interaction. New translation functions had to be added or refined. We expect the issue to be resolved with the help of more experience and data we get from other applications. (2) Some properties needed additional information beside the catalog. (3) Push translations can lead to ambiguous definitions of requirements. The current implementation ("take what's defined first") is not always satisfying. A solution for the issue is either human interaction or a precedence order. (4) The number of resulting requirements at the end of the deduction process becomes extremely large, as all deducible requirement types are inferred without additional reasoning.

VI. Conclusions

In this paper, we proposed a novel requirement definition process for wireless sensor networks to bridge the semantic gap between application requirements as they can be expressed by end-users and technical terms and constraints, as they are needed for the WSN design process. As a first step, a new requirement catalog assists the requirement analysis for WSN applications. This catalog is specifically designed with the end-user in mind. It allows the easy and complete specification of different WSN missions. Since a more detailed technical specification is required for the WSN design, we proposed, as a second step, a requirement expansion methodology. The demonstration of the

Table II
REQUIREMENTS FOR THE FWA DEMONSTRATOR

Category	Requirement Dimensions	Value
Mission	sensors	flow rate, pressure
	sampling interval	30 s
	coverage	points of interest
	number of measurement points	5
	accuracy of spatial correlation	*n/a (exact positions are known)*
	accuracy of temporal correlation	60 s
	mode of operation	monitoring, event detection *(optional)*
	mobility of sensors	static
	mobility of observer	static
Operation Environment and Deployment	dimensionality	two-dimensional
	size	17 500 m × 10 m
	environment conditions	mixed
	radio interference level	high
	radio regulations	Germany/Europe
	available infrastructure	power grid *(sensor nodes, only)*
	mode of deployment	manual
	time-frame of deployment	one-time
	accessibility	limited accessibility
Performance and Dependability	lifetime	3 months
	availability	98 %
	channel dependability	98 %
	response time	2 s
Security	eavesdropping resistance	low
	tampering resistance	low
	denial of service resistance	low
	access control	restricted, none
	stealthiness	none
Development Costs	monetary costs	20 000 €
	development effort	*unknown*

requirement definition process, employed in the design of a new WSN application in the area of critical infrastructure monitoring, showed concepts and benefits of the process, but also exposed potential room for further improvement.

Thus, in the future we will optimize the given catalog and extend the set of requirement expansions based on practical experience and feedback from the community.

ACKNOWLEDGMENT

This work was supported by the German Research Foundation (DFG), grant GRK 1076/3 (TrustSoft), and the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n^o 225186 (WSAN4CIP).

REFERENCES

[1] P. Buonadonna, D. Gay, J. M. Hellerstein, W. Hong, and S. Madden, "TASK: Sensor network in a box," in *Proc. of the 2nd European Workshop on Sensor Networks (EWSN)*, 2005.

[2] B. Greenstein, E. Kohler, and D. Estrin, "A sensor network application construction kit (SNACK)," in *Proc. of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[3] S. Peter, K. Piotrowski, and P. Langendörfer, "In-network-aggregation as case study for a support tool reducing the complexity of designing secure wireless sensor networks," in *Proc. of the Third IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp)*, 2008.

[4] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless micro-sensor network models," *SIGMOBILE Mobile Computing and Communication Review*, vol. 6, 2002.

[5] K. Römer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, 2004.

[6] V. Rocha and G. Gonçalves, "Sensing the world: Challenges on WSNs," in *Proc. of the IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, May 2008.

[7] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proc. of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2006.

[8] D. G. Abraham, G. M. Dolan, G. P. Double, and J. V. Stevens, "Transaction security system," *IBM Systems Journal*, vol. 30, no. 2, 1991.

[9] S. H. Weingart, S. R. White, W. C. Arnold, and G. P. Double, "An evaluation system for the physical security of computing systems," in *Proc. of the Sixth Annual Computer Security Applications Conference*, Dec. 1990.

[10] P. Zhang, C. Sadler, S. Lyon, and M. Martonosi, "Hardware design experiences in ZebraNet," in *Proc. of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[11] WSAN4CIP project, "Deliverable D1.1," 2009. [Online]. Available: http://www.wsan4cip.eu/fileadmin/public_documents/Deliverables/WSAN4CIP_D1.1_Def.of.parameters.pdf