The System Development Lifecycle – Learning from a Sensornet Review

TIK Report 283

Matthias Woehrle, Jan Beutel, Lothar Thiele
Computer Engineering and Networks Lab, ETH Zurich
8092 Zurich, Switzerland
`matthias.woehrle@tik.ee.ethz.ch`

Abstract— **Wireless Sensor Networks (WSNs) have been promised a bright future and economic potential. Since the vision of 'Smart Dust', the field has largely progressed: Miniaturization, integration and advances in MEMS technology have significantly stimulated research efforts. While there has been a lot of progress in theoretical work and system optimizations, actual deployments have been non- or underperforming. The realism of the deployments has exposed substantial and serious barriers to initial application ideas, to interdisciplinary research and thus to the adoption of WSN systems. By reviewing the past and present of WSN systems and their development, significant conclusions can be drawn.**

This work presents a review starting from the 'Smart Dust' vision, describing WSN platforms and components. While assessing the current state of the WSN system development lifecycle, novel ideas for WSN research are discussed. The ideas and suggestions presented in this work serve as a basis for future discussions and support the advancement of WSNs.

I. INTRODUCTION

The starting point of WSNs can be attributed to Kahn et al.'s [1] vision of "Smart Dust" where minuscule sensors are deployed without particular effort (e. g. dropped from a plane) instrumenting areas at large scale. The initial idea of optical communication, although offering superior energy efficiency, has been generally abolished due to the fact that most implementations cannot adequately satisfy line-of-sight constraints. Similarly, the wealth and complexity of issues on the system level often only unraveling to their full magnitude in practice, have lead to substantial efforts being made in areas unanticipated by the early visionaries. As an example, experience has shown that exact sensor node placement is highly critical not only for sensing but also for wireless communication and the scavenging of energy from the environment. Details often deemed trivial such as node placement [2], node protection (e. g. protect against bird droppings [3]) or wireless propagation closely over ground [4] have rendered system design and development a very tedious, extremely complex and often error-prone process requiring careful planning. With respect to the "Smart Dust" vision the numerous intricate details merely concerning placement render the initially perceived airborne deployment practically infeasible.

WSNs are expected to become a radical innovation similar to the internet. WSNs should be operationable and manageable for laymen. Functionality for a high ease-of-use access just as the browser has done for the internet is a vital factor while the underlying architecture needs to be ultimately reliable. Best-effort approaches with superior service levels as used in mobile telephony need to guarantee basic reliability and quality of service to provide satisfiable results to the end user. Nevertheless, the design of a WSN system with its complex interactions and system intricacies today requires in-depth knowledge and cannot be performed by persons from other fields that want to employ WSN technology in their own domain. As the field of WSN continues to open up new application spaces, interaction and inter-disciplinarity has to be supported. A framework has to be provided to determine detailed requirements of a platform, in order to be able to aggressively optimize the system and provide users with a usable and satisfactory system solution. The level of detail for accessing, programming, designing or even optimizing system characteristics needs to be adaptable for a wide range of expertise of the system users.

A. Application Areas

Starting from initial applications and funding in the military sector, with applications such as detection of intruders, there have been numerous fields attracted to WSN technology. Applications for the detection of fires or other emergency situations have emerged. A better understanding of environmental phenomena by detailed monitoring allows for modeling the environment and forecasting. Thus environmental monitoring has been a fertile field for WSNs concerning fauna [5] and flora [6]. Smart building and office monitoring solutions are the indoor equivalent of environmental monitoring. Structural monitoring [7] for bridges, e. g. using acoustic sensors to monitor stress and micro-cracks is another active research field.

Roemer et al. [15] provides a broad overview of WSN applications to present the design space of WSNs. Table I approaches the design space differently by classifying WSNs applications based on their purpose and the resulting requirements. Long-term monitoring typically focusses on sustainable operation for a long system lifetime. Continuous measurements are periodically sent to base stations. Scheduling and coordinated message transport allows for aggressive optimizations of power consumption and low duty-cycles. The environment stresses the sensor nodes by changing ob-

WSN class	Sector	Environment	Example
Long-term monitoring	Environmental	Fauna	Great Duck Island [5]
		Flora	Grape Monitoring [8]
		Hostile Conditions	Glacier [9]
	Urban infrastructure	Bridges	Bridge monitoring [7]
		Buildings	Sick buildings, HVAC [10]
Anomaly detection	Military	Intrusion	ExScal [11]
	Public	Fire	Fire alarm detectors [12]
Mobile	People	Health	Vital signs [13]
		Assistance	Body Glove [14]

TABLE I

WSN SYSTEM CLASSIFICATION

structions, unpredictable weather conditions or a generally hostile climate such as on a mountain. Urban infrastructure differs in that the nodes are typically better protected, but the human impact e. g. concerning mobile telephony or WLAN has a considerable effect on the network operation. Anomaly detection focus on a timely notification on events such as a fire or an intruder. Responsiveness of the network and reliability of the notification are of utmost importance. Coverage of the phenomenon area must be established possibly resulting in denser deployments.

B. WSN Deployment

Numerous WSNs deployments failed to perform [5], [16] or to work at all [6], albeit ingenious engineering efforts. A critical factor is that due to the novel environments and tight system integration within, detailed models necessary for an understanding of the requirements are not available. As an example in environmental monitoring, seasonal changes and its effect on plant growth heavily influence sensed data, harvestable energy and the properties of communication [17]. This results in many system designs having to rely on either no, weak or simplified assumptions. The realism of failing and underperforming deployments has heavily influenced system design. Detailed provisions and focused debug, maintenance and monitoring enhancements improve WSN reliability and performance. Researchers try to actively attack these problems by integrating sophisticated mechanisms into their designs relying on in-depth knowledge [18] and increased visibility by design [19]. Various tools attack the debugging at the deployment site by listening to WSN traffic [20], remote access to the sensor nodes [21], visual monitoring [22] or integrating source-level debugging mechanisms for remote debugging [23]. Nevertheless, WSN design today is tedious and requires attention to intricate details.

C. WSN Systems in the Real World

Currently, there seems to be a chasm between theoretical work and actual real-world installations [16]. Misconceptions persist as researchers incorporated overly simplified assumptions into their models, e. g. for communication [4], in order to cope with the complexity of non-deterministic behavior found in WSNs.

Tanenbaum et al. [24] suggest that due to the realism of the deployment, radio and sensor characteristics, the applications targeted initially are not technically feasible. He predicts that they neither will be technically nor economically feasible with the currently followed WSN design paradigms.

D. WSN Future and Moore's Law

The sensor nodes have significantly benefited from Moore's Law integrating minuscule microprocessors, memory and radio chips at a reasonable price. Looking at the future there seem to be two trends concerning computing and shrinking feature sizes: *scale-up* and *scale-out*.

Mote-class devices may be designed with an even smaller form factor and reduced price by a *scale-out* approach, allowing significantly tighter integration with the environment i. e. using more nodes for instrumentation. However, for many applications high-fidelity sensors may cost magnitudes more than the integrated device. Considering the efforts necessary for software development and deployment the cost of the mote-class device is just a minor item of the total cost of ownership. Similarly, the form factor for the integrated silicon is not necessarily a selling-point, since other components (batteries, connectors) dictate a minimum scale for a WSN platform.

When looking at a *scale-up* approach, sensor nodes may be designed to be more powerful, allowing more memory for storage and increased processing power. However, power consumption in sub-micron technologies does not scale well with feature size due to considerable leakage current in nanometer technologies. Thus the increased functionality has to be paid by additional energy provisioning. Nevertheless, higher performance microprocessors and larger memory may be used to handle the burstiness of applications while sleeping most of the time to stay in the energy bounds. This requires components that feature aggressive power-down modes to alleviate for the leakage problems in sub-micron technologies. However, not all applications require aggressive energy optimization and in select cases may work with powerful platforms, especially when possibilities for energy harvesting [25] are abundant.

Motivated by the selected observations discussed in this section this paper reviews the current practice in WSN system design. The shortcomings in the prevailing development paradigm and the resulting platforms are discussed identifying the key problems in the system development lifecycle. Basis for the discussion is a review of WSN platforms and components. In the closing discussion, we focus on suggestions for improvement on selected items, we see as most critical such as the pragmatic use of prototyping platforms, integrated design, development and validation tools and novel research on recently emerging classes of WSN specific tools for validation.

II. WSN PLATFORMS AND COMPONENTS

A. *WSN Nodes - A Simple Picture*

Mote-class devices instrument a phenomenon in a specific target area. Data is collected and aggregated at sink or root nodes which typically have additional processing, energy and storage resources and provide the ability to connect to a secondary network topology acting as gateways.

Looking simplistically onto a single WSN node, there are a few distinct services the node provides:

- Sensors that collect some data and ADCs to convert into the digital domain and possibly generate threshold interrupts
- A timer subsystem for periodic events with differing granularity
- Interrupt processing
- Simple data processing
- Communication capabilities using a protocol stack

B. *WSN Hardware*

There have been various different WSN hardware platforms used in research, but most of these mote-class platforms are very similar in design, using the same standard parts, e. g. the Atmel AVR, the TI MSP430 or lately the 8051 architecture [26]. The commonly used platforms are built of readily available components integrated onto a board with a small form factor. Due to node cost requirements only Commodity-Off-The-Shelf components are selected. The cost of application-specific hardware renders custom design for individual projects economically infeasible.

The processing power of the 8 or 16 bit microcontrollers is fairly limited, comparable to that of the first PC's in the early 80's. Application-specific protocols exploit low power states of the hardware components to drastically minimize the power consumption by heavy duty cycling. Low power radios provide limited transmission range. They are either bit-stream oriented or packet-based. On-chip data memory allows for temporarily storing measurements. Larger flash integration is supported by Moore's Law concerning price and area and by sophisticated power saving techniques allowing to integrate an increasing amount of memory without inferring a large overhead on the energy source [27].

A second type of platforms has emerged out of the requirement that each deployment needs some form of access to transfer the aggregated data from the sensor domain into the typically wired domain of the users. The need for such gateways has resulted in more powerful nodes, with increased processing power (32-bit CPUs), larger memory as well as energy resources. If used in combination with mote-class devices, they are typically used as cluster-heads in the resulting heterogeneous architectures resulting in a shift of interest towards the specific problems of such heterogeneous tiered architecture [28].

It was claimed then, that the use of sophisticated load balancing and in-network processing would render the system design and management more structured and flexible and more extendible. Today, little of this claim is still visible and most research focuses on rather complex architecture but a simpler design process. The ability to achieve a considerable bandwidth to a data sink lead to a decision towards multiple sinks or intermediate tiers with individual nodes arranged in possibly multiple collaborative trees [29].

In order to aid in the design, a fast-prototyping platform with programmable hardware and flexible memory is not available. However in safety-critical embedded system design, e. g. for avionics or automotive applications, rapid prototyping platforms are typically used for exploration and early tests before the actual hardware is available.

C. *WSN Software*

The availability of standard platforms and economical restrictions of custom hardware design has resulted in an increased focus towards embedded software. Application-specific optimizations are pushed towards the software design, in particular the communication protocols rendering protocol design a lively field of research. As one example, numerous MAC protocols are available addressing a large variety of requirements. An overview of MAC protocols for WSNs is presented in Langendoen [30].

An interesting factor for WSNs is that in a deeply embedded system with very tight energy constraints, there is a very close interaction and dependency of individual protocol layers. Optimization of the stack requires a comprehensive analysis by studying cross-layer dependencies.

Reliability concerns as typically found in communication protocols favor simple designs without relying on environmental or platform assumptions allowing the system to run sustainable even under deviating conditions. However, the exploitation of such assumptions is the basis of aggressive optimizations used in WSN software design to minimize the power consumption.

As the applications for WSNs differ, so are the requirements and research efforts. The operating system as well as middleware services are common grounds for a large number of domains resulting in a research focus.

The operating systems for WSNs are designed very lean, as the underlying architecture does not provide sophisticated mechanisms, such as address translation and access protection as encountered in today's desktop microprocessors. Instead these OS's follow the low-power design paradigm and restrict themselves to minimal capabilities, i. e. , interrupt handling and simple computation such as packet processing.

The dominantly used open-source operating system TinyOS [31] is a prominent example: Using the nesC programming language, a C dialect allowing the componentization of software similar to hardware description languages, TinyOS provides a two-level concurrency model with interrupt routines, which can be either preemptive or non-preemptive, and a separate non-preemptive task queue. The intricacies of almost bare-metal programming w. r. t. interrupt handling and masking, and the familiarity with typical desktop OS's has also stimulated research in operating systems providing advanced mechanisms to use lightweight threading such as Contiki [32] or even provide OS protection and virtual memory support as in t-kernel [33].

D. Design Criteria

1) Optimization Criteria: Due to the tight resource constraints of embedded systems, software components and according resource usage have to be optimized. One of the most fundamental issues is the constraint of energy supply. For such heavily energy constrained systems, power consumption needs to be meticulously optimized.

One of the basic facts for WSNs is that computation is cheaper than communication, i. e. , $E_{comp} \ll E_{comm}$. Thus processing of data fundamentally consumes less power than communication of data [34]: While computation is not for free, sending 1kb of information across 100 meters consumes the amount of energy required for executing millions of instructions on a general-purpose microprocessor. Energy for transmitting and receiving are comparable in energy consumption. Burri et al. [35] present radio power consumption for different states, showing that even idle listening is comparable in power consumption to receiving or transmitting and powering off the radio thus renders significant returns in energy savings. Several approaches have emerged trading off on-chip computation for communication.

2) Communication Intricacies: Radio communication is a significant characteristic of WSNs based on two factors: unreliability and power consumption. Unreliable communication requires each node to incorporate countermeasures for failed transmission due to interferences and collisions. The ether as a broadcast medium necessitates arbitration of the broadcasting nodes to avoid collisions. Noise, anisotropies and multi-path effects deteriorate packet reception rates.

As communication is expensive in terms of energy costs, a general goal is to minimize the amount of messages to be transmitted or received. On an application level, an event-based or scheduled communication approach is preferable over energy intensive data polling. Aggregation of data [36] in data-centric routing allows for reducing the amount of messages. Such approaches must consider trade-offs concerning the accuracy, periodicity and latency requirements of the sensor data. Energy may also be wasted in communication due to overhearing, idle listening, collisions and control packet overhead. Considering the small amount of payload the sensor nodes typically transmit, the control overhead concerning the maintenance and protocols are significant.

The cost of powering on the radio, e. g. for carrier sense renders synchronous, slotted approaches with low-power duty cycles attractive for low data rate applications of the WSN domain. Moreover, costs for scheduling are typically offset by communication benefits, but global synchronization is intricate.

An example of a comprehensive low power communication stack, in this case with a slotted TDMA approach for ultra low-power environmental monitoring, is Dozer [35]. Such highly integrated protocol stacks are very complex, e. g. due to the need to include the routing tree in the schedule computation.

Asynchronous low-power listening protocols rely on the receiver sleeping and only periodically sampling the channel. The sender may send very long wake-up preambles to match a receiver listening [37] or use coordination to significantly reduce the preamble length [38].

3) Organization and Maintenance: Since WSN deployments should run with minimal need for maintenance, self initialization, organization and maintenance is of utmost importance. Especially in inaccessible terrains, network health monitoring [18], [39] support the maintenance and health of the deployment. Reprogramming is another vital service for maintaining a deployed system (cf. Sec III-B.1).

4) The Omnipresent Future: WSNs may become ubiquitous and omnipresent if the technical feasibility and its application benefits provide the economic returns envisioned. In this case, such a ubiquitous system needs to be available requiring safe and secure operation in a possibly detrimental environment. Perrig et al. [40] discuss that there are numerous susceptibilities such as physical tampering. Traditionally used security measures cannot be used for sensor networks due to the resource shortage. However, accepting a small energy overhead a general purpose security protocol has been shown to be feasible [41].

A fundamental problem of WSN nodes are Denial-of-Service (DoS) attacks. Transmission may be maintained in presence of jammers with spread-spectrum radios or frequency-hopping. However, spam messages transmitted by attacking jammers snooped on a node drain its battery. Any mote with a receiver is susceptible to such DoS-Energy attacks. One possible solution is to allow sensing devices only to transmit data as currently available in light switch products [42]. While this does not allow for multi-hop networks, a hierarchical network may be designed, where the lowest sensing layer only transmits its data, but in turn is resilient against tampering. Thus, DoS-Energy attacks would only be possible on nodes a higher tier, where larger energy resources would assure an acceptable lifetime to detect the anomaly and remove a malicious jammer.

III. WSN DEVELOPMENT LIFECYCLE

The focal point of the design process is a specification, collecting all functional and non-functional requirements of the system. The specification is an agreement between the customer or user and the developer. It is the core document for the design and development cycle and used to verify and validate the systems function and performance. It has been shown that efforts spent on specification and the identification of the requirements pays off over the whole system development lifecycle [43].

A. WSN as a Research Tool

WSNs are used by many research groups in very different fields. The wireless and ubiquitous nature of WSNs provide new and unforeseen properties to the researchers in the field, which are striving for a better understanding and accurate models of the phenomenon to be observed. Coming from the 'Smart Dust' vision, researchers are mostly focussed on deeply embedded and ubiquitous sensor nodes. While some researchers see the emergence of more powerful platforms for the complete system for the near future, no comprehensive work has yet addressed the powerful notion of using a powerful platform as a research enabler. While this may boost the interdisciplinary research, the work on the smaller, deeply integrated systems is orthogonal, since the availability of a comprehensive specification allows for optimization of the system, rendering tiny platforms as WSN components as a viable and cost-effective approach. Cost is a major factor for WSNs to become successful and omnipresent in industrial products.

For interdisciplinary research between other scientific fields and the WSN community, there is a significant challenge in building functioning, performing and reliable systems. The reason is that the system may not be completely specified at design time. Researchers typically do not know the quantity of required sensors, the sampling period, nor do they have a clear and definite description of the system environment. Thus researchers look for the culmination of an ultimate sensor node for an initial prototype: feature-rich, everlasting, resilient and reliable. Before restricting to limited data researchers are looking for a rich data set to better understand the observed phenomenon and to be able to derive an accurate model.

Some projects have already shown that prototyping is a viable approach [44]. Starting from an over-provisioned, feature rich prototype, collected real-world data may be used to minimize sensory inputs [14]. The WSN community has not addressed the need for platforms targeted prototyping and system exploration, since the primary goal is to show the applicability and the feasibility of tiny, low-performance sensor nodes.

Some technologies like the Sun Spots [45] combine a more powerful processor with a low power microcontroller for controlling the duty cycles for deep sleep states. This might be a first step towards a prototyping platform, since it allows to employ the standard Java design methodology along with available tools and support.

B. Development Process

The design and development is largely supported by an environment providing support in the form of tools and a knowledge base. Thus a Sun Spot developer benefits from Java tools such as Eclipse or the JUnit [46].

One particular aspect for wireless distributed systems is the update of the sensor node code. This method is particularly important for the development process, but may also be used in trial or the actual deployments.

1) Code Updates: WSN nodes typically feature a microcontroller with a harvard architecture directly connected to a Flash EEPROM, from which code is executed from. Code cannot be executed from RAM, so in order to make any changes, the code image in the ROM has to be modified. This approach is very well suited for the embedded applications which are using low cost, low power, low resource microcontrollers. However, this approach is not suited for multiple updates or patches as it is the case for the development cycle or a research environment where the application and its requirements are still in flux. Considering the distributed nature of sensor nodes and sending code updates over the wireless channel, each individual update generates significant energy costs. Research in WSNs have approached the question of code updates with different approaches:

- Code image distribution for TinyOS and update mechanisms [47]
- Dynamically loaded applications such as in Contiki [32]
- Virtual Machines as proposed for Mate [48]

Different mechanisms trade off between flexibility and update frequency for energy efficiency. Dunkels et al. [49] discuss a comparison of dynamically linked and loaded ELF objects, versus a customized VM or a standard Java Virtual Machine. Nevertheless, the design and development stage with frequent code updates has to be differentiated to a deployment not allowing or restricting code updates. Nevertheless, code updates may also be used for deployed systems. A comprehensive overview on code updates for WSNs is provided by Brown et al. [50].

C. Design Level

Software design has profited from the introduction of higher level languages, thereby raising the design level for the programmer, allowing for increased productivity, readability and understandability of software code. Even for embedded platforms, designers typically do not design in assembler code, but use an intermediate language like C and compile for their target platform. Assembler is mainly used for manually optimizing critical sections for performance or resource use. Abstraction is traded-off against performance where needed. WSNs using embedded components with tight resource constraints are typically monolithically designed within an OS (framework). Researchers have proposed the use of Virtual Machines or Abstract State VM (ASVM) [51] or the Tenet Building Blocks [52]. This allows for designing the system on a higher level, to increase the understandability and productivity.An integrated approach like ASVM, where crucial constructs may still be implemented as efficient primitives promises the benefits of both optimization and abstraction.

High-level design of the event-driven systems may be supported by higher-level modeling concepts like Attributed State Machines [53], where the design is specified using state machines to alleviate problems concerning control flow, manual stack management and application validation.

On a system level, approaches like Kairos [54] try to express the global behavior of the comprehensive sensor network. Middleware layers try to hide some of the complexities of distributed programming to facilitate application programming. An example in the context of tracking the environment is EnviroTrack [55]. It provides an abstraction on top of TinyOS to allow for formulating the detection of an event occurrence based on localized sensor readings and tracking as it moves through the instrumented environment. The underlying architecture provides services such as clustering of event readings by leader election or handoff.

The design level for an application should be chosen based on a trade-off of performance, which typically requires lower level access in the system and ease-of-use for reliable, reusable and understandable code typically written on a higher level of the system. Design for test and debugging is also a critical factor [56]. ASVM for example allows for TinyOS integration of selected application primitives, while the application is written on top of the customized virtual machine. Further interesting approaches are the modeling of the complete software system with a formal model, e. g. with hybrid automata [57].

The design and the validation process can be enhanced by using of assertions on interfaces as described with interface contracts for TinyOS [58] allows for verifying function implementation and usage. Memory safety mechanisms [59] provide support at compile- and runtime without incurring a large overhead. Safe TinyOs helps avoiding, detecting and catching problems with memory, which are typically very hard to find, since the manifestation is sporadic and the behavior of failing nodes is byzantine.

Over-provisioned or configurable hardware may be used as a test vehicle for initial debugging and profiling of an application. Feasibility studies and technology outlooks may be performed even when details for the data acquisition are not sufficiently available.

Initial prototypes may be focused on a more limited set of assumptions on the hardware and more detailed data aggregation. WSN users want to acquire a rich set of samples to determine an accurate or satisfying model of the observed phenomenon. With the according rich data set, outlooks may be performed where trade-offs between data quantity, data quality, timeliness and lifetime may be explored. Based on the resulting (reduced) model and according abstractions, the underlying WSN infrastructure may be further constrained. These constraints can be used to further optimize the system and the sensor nodes.

D. WSN validation

System validation has relied on different test platform types: simulation, emulation and testbeds. There is wide set of simulators. These include well-known simulators with their origin in traditional (wireless) network design like Ns-2 or GloMoSim. Libraries have been developed by the community to include the domain-specific aspects of WSN, particularly the radio. They offer scalability, but typically provide a higher ease-of-use and accuracy for other domains than for WSN. A simulator targeted for WSNs is Castalia[60] based on OMNet++, which is sensor node agnostic and targeted for an initial validation of sensornet algorithms.

WSN specific simulators have been established, like EmSim [61] and TOSSIM [62], which focus on using the actual target code and link it with simulation libraries to run on a host computer for simulation.

Simulation deficiencies and realism of deployment challenges has triggered an increased interest in testbed implementations for Motelab [63] or the Deployment Support Network [64]. These are installations at the department building or on the university campus to better grasp deployment characteristics. Researchers hope to accommodate better for the non-deterministic nature of the wireless channel by testing their code on a testbed. Other significant parameters in the system space, such as solar energy scavenged for sustainable operations and others effect of a outdoor environment, require either physical stimulation and control or a representative outdoor testbed such as Trio [3]. Nevertheless, testbed characteristics for radio communication, topology, or harvestable energy may still differ considerably to the actual deployment site.

Researches have proposed field trials [65] in order to attain deeper insight into WSNs. Turau et al. discuss that such prototypical deployments are expensive, but nevertheless provide invaluable information about the environment and actual system execution on a large-scale, especially as systematic validation and testing methodologies and tools are largely missing.

Cycle-accurate simulation has been proposed [66], which allows the designer to run the compiled code with increased visibility. However, the intricacy of WSN system design for an experienced embedded designer is not the assembly code for a limited 8-bit microcontroller, but the complex interplay of a multitude of embedded devices in a non-deterministic and typically insufficiently characterized environment. On the debugging side, Clairvoyant[23] offers a source level distributed debugger with wireless, remote access, which provides novel, WSN specific commands as well as standard debugger commands.

Hybrid solutions (EmStar [61]) offer some alleviation to the problem, but need to be carefully designed with a considerable one-time cost. Although multiprocessor computers are available at reasonable cost providing dominant computation power 2 to 3 orders of magnitude stronger than the actual target platform, in-depth system simulators including detailed node and radio models are not available to this date.

Additionally, with the help of the VipTos [67], a Ptolemy II descendant targeted for TinyOS applications and integrated with TOSSIM, the designer is offered a rich framework allowing system modeling and testing with the help of the different integrated models of computation.

TUnit[68] is a unit-testing framework for TinyOS software and available as a contributed project from the TinyOS repository. Unit tests are specified and run on actual hardware platforms. Regression testing on the TinyOS core libraries underlines the significance of unit testing for validation.

The distributed testing framework for WSNs presented in Woehrle et al. [69] supports continuous testing throughout the design cycle by exploiting the ability to design and test on different abstraction levels with a common test specification, promoting regression testing and test integration into a periodic build process. COOJA [70] integrates simulation on different abstraction levels into a integrated framework.

IV. DISCUSSION

Wireless Sensor Networks is an interesting and challenging research field requiring novel approaches and ideas due to the combination of different fields with widely differing properties, constraints and paradigms. We presented a review of WSNs starting from the 'Smart Dust' vision, presenting WSN platforms and components and the system development lifecycle.

We described new view points for researchers in the field by highlighting approaches from other fields, e. g. concerning current issues in prototyping and the according prototyping process in automotive electronic system designer. Prototypical installations in the beginning of the project employing approaches from rapid prototyping may provide detailed environmental information without incurring the considerable cost of a real deployment.

We discussed the fundamental properties of WSN and their implications on the system development lifecycle, i. e. the design, development and validation of of WSNs. We argue that instead of deploying a near-completion system, research should be focussed on systemic approaches for the pre-deployment stage. Research on design and test platforms has yielded a considerable number of significant contributions considering methodologies and tools.

However, missing are still methodologies for the validation process and the integration of design, development and validation tools in order to create a comprehensive system development environment. Just as for the WSN system itself, a system development environment is more than its individual components. An integration of validation tools into the development lifecycle increases the ease-of-use considerably and is required for an adoption by the developers.

REFERENCES

[1] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next century challenges: mobile networking for smart dust," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM Press, 1999, pp. 271–278.

[2] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Coverage problems in wireless ad-hoc sensor networks," in *INFOCOM*, 2001, pp. 1380–1387.

[3] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler, "Trio: enabling sustainable and scalable outdoor wireless sensor network deployments," in *Proc. 5th Int'l Conf. Information Processing Sensor Networks (IPSN '06)*, 2006, pp. 407–415.

[4] D. Kotz, C. Newport, and C. Elliott, "The mistaken axioms of wireless-network research," Dartmouth College Computer Science, Tech. Rep. TR2003-467, 2003.

[5] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," in *Proc. 1st European Workshop on Sensor Networks (EWSN 2004)*, 2004, pp. 307–322.

[6] K. Langendoen, A. Baggio, and O. Visser, "Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture," in *Proc. 20th Int'l Parallel and Distributed Processing Symposium (IPDPS 2006)*, 2006, pp. 8–15.

[7] W. Xu, P. Bodik, and D. Patterson, "A flexible architecture for statistical learning and data mining from system log streams," in *Workshop on Temporal Data Mining: Algorithms, Theory and Applications at The Fourth IEEE International Conference on Data Mining (ICDM'04)*, 2004.

[8] R. Beckwith, D. Teibel, and P. Bowen, "Unwired wine: sensor networks in vineyards," in *Sensors, 2004. Proceedings of IEEE*, 2004, pp. 561–564.

[9] K. Martinez, R. Ong, and J. Hart, "Glacsweb: a sensor network for hostile environments," in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, 2004, pp. 81–87.

[10] N. Ota, D. Hooks, P. Wright, D. Auslander, and T. Peffer, "Poster abstract: wireless sensor network characterization - application to demand response energy pricing," in *Proc. 1st ACM Conf. Embedded Networked Sensor Systems (SenSys 2003)*, 2003, pp. 334–335.

[11] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler, "Design of a wireless sensor network platform for detecting rare, random, and ephemeral events," in *Proc. 4th Int'l Conf. Information Processing Sensor Networks (IPSN '05)*, 2005, pp. 497–502.

[12] M. Strasser, A. Meier, K. Langendoen, and P. Blum, "Dwarf: Delay-aware robust forwarding for energy-constrained wireless sensor networks," in *Proceedings of the 3rd IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS 2007)*, June 2007, pp. 64–81.

[13] J. Stankovic, Q. Cao, T. Doan, L. Fang, Z. He, R. Kiran, S. Lin, S. Son, R. Stoleru, and A. Wood, "Wireless sensor networks for in-home healthcare: Potential and challenges," *In High Confidence Medical Device Software and Systems (HCMDSS) Workshop*, April 2005.

[14] L. A. Rachel King, "An hmm framework for optimal sensor selection with applications to bsn sensor glove design," in *Proc. 4th IEEE Workshop on Embedded Networked Sensors (EmNetS-IV)*, 2007.

[15] K. Römer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54–61, Dec. 2004.

[16] J. Choi, J. Lee, M. Wachs, and P. Levis, "Opening the sensornet black box," in *Proc. Int'l Workshop Wireless Sensornet Architecture (WWSNA '07)*. ACM Press, New York, 2007, p. to appear.

[17] P. K. Haneveld, "Evading murphy: A sensor network deployment in precision agriculture," June 2007. [Online]. Available: http://www.st.ewi.tudelft.nl/~koen/papers/LOFAR-agro-take2.pdf

[18] S. Rost and H. Balakrishnan, "Memento: A health monitoring system for wireless sensor networks," in *Proc. 3rd IEEE Communications Society Conf. Sensor, Mesh and Ad Hoc Communications and Networks (IEEE SECON 2006)*, 2006.

[19] M. Wachs, J. I. Choi, J. W. Lee, K. Srinivasan, Z. Chen, M. Jain, and P. Levis, "Visibility: A new metric for protocol design," in *Proc. 5th ACM Conf. Embedded Networked Sensor Systems (SenSys 2007)*, 2007.

[20] M. Ringwald, K. Römer, and A. Vitaletti, "Passive inspection of sensor networks," in *Proc. 3rd IEEE Int'l Conf. Distributed Computing in Sensor Systems (DCOSS 2007)*, Santa Fe, New Mexico, USA, June 2007.

[21] K. Whitehouse, G. Tolle, J. Taneja, C. Sharp, S. Kim, J. Jeong, J. Hui, P. Dutta, and D. Culler, "Marionette: using RPC for interactive development and debugging of wireless embedded networks," in *Proc. 5th Int'l Conf. Information Processing Sensor Networks (IPSN '06)*. ACM Press, New York, 2006, pp. 416–423.

[22] H. Liu, L. Selavo, and J. Stankovic, "Seedtv: deployment-time validation for wireless sensor networks," in *Proc. 4th IEEE Workshop on Embedded Networked Sensors (EmNetS-IV)*, 2007, pp. 23–27.

[23] J. Yang, M. L. Soffa, L. Selavo, and K. Whitehouse, "Clairvoyant: A comprehensive source-level debugger for wireless sensor networks," in *Proc. 5th ACM Conf. Embedded Networked Sensor Systems (SenSys 2007)*, 2007.

[24] A. S. Tanenbaum, C. Gamage, and B. Crispo, "Taking sensor networks from the lab to the jungle," *Computer*, vol. 39, no. 8, pp. 98–100, 2006.

[25] P. Corke, P. Valencia, P. Sikka, T. Wark, and L. Overs, "Long-duration solar-powered wireless sensor networks," in *Proc. 4th IEEE Workshop on Embedded Networked Sensors (EmNetS-IV)*, 2007, pp. 33–37.

[26] TinyOS 8051 Working Group, "Tinyos 8051 working group — tinyos8051wg," September 2007. [Online]. Available: http://www.tinyos8051wg.net/

[27] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy, "Ultra-low power data storage for sensor networks," in *Proc. 5th Int'l Conf. Information Processing Sensor Networks (IPSN '06)*, 2006, pp. 374–381.

[28] O. Gnawali, K.-Y. Jang, J. Paek, M. Vieira, R. Govindan, B. Greenstein, A. Joki, D. Estrin, and E. Kohler, "The tenet architecture for tiered sensor networks," in *Proc. 4th ACM Conf. Embedded Networked Sensor Systems (SenSys 2006)*. New York, NY, USA: ACM Press, 2006, pp. 153–166.

[29] P. C. L. Mottola and G. P. Picco, “Efficient routing from multiple sources to multiple sinks in wireless sensor networks,” in *Proc. 4th European Workshop on Sensor Networks (EWSN 2007)*, 2007, pp. 34–50.

[30] K. Langendoen, *Medium Access Control in Wireless Networks*. Nova Science Publishers, to be published, ch. Medium Access Control in Wireless Sensor Networks.

[31] P. Levis, D. Gay, V. Handziski, J.-H.Hauer, B.Greenstein, M.Turon, J.Hui, K.Klues, C.Sharp, R.Szewczyk, J.Polastre, P.Buonadonna, L.Nachman, G.Tolle, D.Culler, and A.Wolisz, “T2: A second generation os for embedded sensor networks,” Telecommunication Networks Group, Technische Universität Berlin, Tech. Rep. TKN-05-007, Nov. 2005.

[32] A. Dunkels, B. Grönvall, and T. Voigt, “Contiki – a lightweight and flexible operating system for tiny networked sensors,” in *Proc. 1nd IEEE Workshop on Embedded Networked Sensors (EmNetS-I)*, 2004, pp. 455 – 462.

[33] L. Gu and J. A. Stankovic, “t-kernel: providing reliable os support to wireless sensor networks,” in *Proc. 4th ACM Conf. Embedded Networked Sensor Systems (SenSys 2006)*. New York, NY, USA: ACM Press, 2006, pp. 1–14.

[34] G. Pottie and W. Kaiser, “Wireless integrated network sensors,” *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, May 2000.

[35] N. Burri, P. von Rickenbach, and R. Wattenhofer, “Dozer: ultra-low power data gathering in sensor networks,” in *Proc. 6th Int’l Conf. Information Processing Sensor Networks (IPSN ’07)*, Apr. 2007, pp. 450–459.

[36] B. Krishnamachari, D. Estrin, and S. B. Wicker, “The impact of data aggregation in wireless sensor networks,” in *Proc. of the 22nd Int’l Conference on Distributed Computing Systems (ICDCSW ’02)*, 2002, pp. 575–578.

[37] W. Ye, J. Heidemann, and D. Estrin, “An energy-efficient MAC protocol for wireless sensor networks,” in *Proc. 21st Ann. Joint IEEE Conf. Computer Communication Soc. (Infocom 2002)*, June 2002, pp. 1567–1576.

[38] A. El-Hoiydi and J. Decotignie, “WiseMAC: An ultra low power MAC protocol for multi-hop wireless sensor networks,” in *Proc. 1st Int’l Workshop Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS 2004)*, 2004, pp. 18–31.

[39] N. Ramanathan, K. Chang, R. Kapur, L. Girod, E. Kohler, and D. Estrin, “Sympathy for the sensor network debugger,” in *Proc. 3rd ACM Conf. Embedded Networked Sensor Systems (SenSys 2005)*. ACM Press, New York, 2005, pp. 255–267.

[40] A. Perrig, J. Stankovic, and D. Wagner, “Security in wireless sensor networks,” *Commun. ACM*, vol. 47, no. 6, pp. 53–57, 2004.

[41] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, “Minisec: a secure sensor network communication architecture,” in *Proc. 6th Int’l Conf. Information Processing Sensor Networks (IPSN ’07)*, 2007, pp. 479–488.

[42] E. GmbH, September 2007. [Online]. Available: http://www.enocean.com/

[43] J. P. Bowen and M. G. Hinchey, “Ten commandments of formal methods ...ten years later,” *Computer*, vol. 39, no. 1, pp. 40–48, 2006.

[44] D. E. Michael Allen, Lewis Girod, “Acoustic laptops as a research enabler,” in *Proc. 4th IEEE Workshop on Embedded Networked Sensors (EmNetS-IV)*, 2007.

[45] Sun, “Wsn platform,” http://www.sunspotworld.com/.

[46] V. Massol and T. Husted, *JUnit in Action*. Greenwich, CT, USA: Manning Publications Co., 2003.

[47] P. J. Marrón, M. Gauger, A. Lachenmann, D. Minder, O. Saukh, and K. Rothermel, “Flexcup: A flexible and efficient code update mechanism for sensor networks,” in *Proc. 3rd European Workshop on Sensor Networks (EWSN 2006)*, 2006, pp. 212–227.

[48] P. Levis and D. Culler, “Maté: A tiny virtual machine for sensor networks,” in *Proc. 10th Int’l Conf. Architectural Support Programming Languages and Operating Systems (ASPLOS-X)*, 2002, pp. 85–95.

[49] A. Dunkels, N. Finne, J. Eriksson, and T. Voigt, “Run-time dynamic linking for reprogramming wireless sensor networks,” in *Proc. 1st ACM Conf. Embedded Networked Sensor Systems (SenSys 2003)*, Nov. 2006.

[50] S. Brown and C. J. Sreenan, “Updating software in wireless sensor networks: A survey,” Department of Computer Science, University College Cork, Ireland, Tech. Rep. UCC-CS-2006-13-07, 2006.

[51] P. Levis, D. Gay, and D. Culler, “Bridging the gap: Programming sensor networks with application specific virtual machines,” UC Berkeley, Tech. Rep. UCB/CSD-04-1343, August 2004.

[52] R. Govindan, E. Kohler, D. Estrin, F. Bian, K. Chintalapudi, O. Gnawali, S. Rangwala, R. Gummadi, and T. Stathopoulos, “Tenet: An Architecture for Tiered Embedded Networks,” CENS, Tech. Rep., November 10 2005.

[53] O. Kasten and K. Römer, “Beyond event handlers: Programming wireless sensors with attributed state machines,” in *Proc. 4th Int’l Conf. Information Processing Sensor Networks (IPSN ’05)*, 2005, pp. 45–52.

[54] R. Gummadi, O. Gnawali, and R. Govindan, “Macro-programming wireless sensor networks using kairos,” in *Proc. of the Int’l Conference on Distributed Computing in Sensor Systems (DCOSS)*, June 2005.

[55] T. Abdelzaher, B. Blum, Q. Cao, Y. Chen, D. Evans, J. George, S. George, L. Gu, T. He, S. Krishnamurthy, L. Luo, S. Son, J. Stankovic, R. Stoleru, and A. Wood, “Envirotrack: towards an environmental computing paradigm for distributed sensor networks,” in *Distributed Computing Systems, 2004. Proceedings. 24th International Conference on*, 2004, pp. 582–589.

[56] J. G. Ganssle, *The Art of Programming Embedded Systems*. Orlando, FL, USA: Academic Press, Inc., 1991.

[57] S. Coleri, M. Ergen, and T. J. Koo, “Lifetime analysis of a sensor network with hybrid automata modelling,” in *WSNA ’02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. New York, NY, USA: ACM Press, 2002, pp. 98–104.

[58] W. Archer, P. Levis, and J. Regehr, “Interface contracts for tinyos,” in *Proc. 6th Int’l Conf. Information Processing Sensor Networks (IPSN ’07)*. New York, NY, USA: ACM Press, 2007, pp. 158–165.

[59] N. Cooprider, W. Archer, E. Eide, D. Gay, and J. Regehr, “Efficient memory safety for tinyos,” in *SenSys ’07: Proceedings of the 5th international conference on Embedded networked sensor systems*, 2007, pp. 205–218.

[60] National ICT Australia, “National ict australia - castalia [home],” September 2007. [Online]. Available: http://castalia.npc.nicta.com.au/

[61] L. Girod, N. Ramanathan, J. Elson, T. Stathopoulos, M. Lukac, and D. Estrin, “Emstar: A software environment for developing and deploying heterogeneous sensor-actuator networks,” *ACM Trans. Sen. Netw.*, vol. 3, no. 3, p. 13, 2007.

[62] P. Levis, N. Lee, M. Welsh, and D. Culler, “TOSSIM: Accurate and scalable simulation of entire TinyOS applications,” in *Proc. 1st ACM Conf. Embedded Networked Sensor Systems (SenSys 2003)*, Nov. 2003, pp. 126–137.

[63] G. Werner-Allen, P. Swieskowski, and M. Welsh, “MoteLab: A wireless sensor network testbed,” in *Proc. 4th Int’l Conf. Information Processing Sensor Networks (IPSN ’05)*, Apr. 2005, pp. 483–488.

[64] M. Dyer, J. Beutel, L. Thiele, T. Kalt, P. Oehen, K. Martin, and P. Blum, “Deployment support network - a toolkit for the development of WSNs,” in *Proc. 4th European Workshop on Sensor Networks (EWSN 2007)*, 2007, pp. 195–211.

[65] V. Turau, M. Witt, and M. Venzke, “Field trials with wireless sensor networks: Issues and remedies,” in *Proc. of the Int’l Multi-Conference on Computing in the Global Information Technology (ICCGI ’06)*, 2006, p. 86.

[66] B. L. Titzer, D. K. Lee, and J. Palsberg, “Avrora: scalable sensor network simulation with precise timing,” in *Proc. 4th Int’l Conf. Information Processing Sensor Networks (IPSN ’05)*, 2005, p. 67.

[67] E. Cheong, E. A. Lee, and Y. Zhao, “Viptos: a graphical development and simulation environment for tinyos-based wireless sensor networks,” in *Proc. 3rd ACM Conf. Embedded Networked Sensor Systems (SenSys 2005)*, 2005, pp. 302–302.

[68] R. R. Corporation, “Tinyos 2.x automated unit testing / tunit,” http://www.lavalampmotemasters.com/, October 2007.

[69] M. Woehrle, C. Plessl, J. Beutel, and L. Thiele, “Increasing the reliability of wireless sensor networks with a distributed testing framework,” in *Proc. 4th IEEE Workshop on Embedded Networked Sensors (EmNetS-IV)*, June 2007.

[70] 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*, Nov. 2006, pp. 641–648.