

Rise of Embedded and Sensor Networks

Janne Riihijärvi* and Petri Mähönen**

*Department of Wireless Networks, Aachen University

**Kackertstrasse 9, D-52072 Aachen, Germany

email: Petri.Mahonen@mobnets.rwth-aachen.de

email: Janne.Riihjarvi@mobnets.rwth-aachen.de

ABSTRACT

Embedded and sensor networks form an exciting and active research area, as well as a set of commercially emerging technologies. The progress on this field has been enabled by advances in the microelectronics, which has increased the capacity of small radios with low price and battery consumption. In this paper we present sketch some aspects of the present-day and foreseeable future research in this field, and line out directions for further studies that we find most interesting.

INTRODUCTION TO SENSORS AND EMBEDDED NETWORKS

In this article we present a review and sketch a possible roadmap on the future of a new and very active research area, namely embedded and sensor networking. We also report in small extent some of our own research as well, and try to point out interesting future research possibilities. As the topic is vast both in terms of the present state-of-the-art, and interesting research opportunities, we cannot claim to present a complete view of the field. Instead, we have attempted to select and discuss some of the research avenues and trends that in our minds seem to offer greatest possibilities for potentially disruptive research. In this way the present article can be seen in part as a sensor and embedded networking extension of [1].

One of the most powerful trends of the recent years in wireless communication has been the introduction of wireless network interfaces into ever smaller devices. A frequently presented vision is that of pervasive sensor networks, tiny devices, down to “SmartDust” scales [2], sprinkled all around our environment, ready to provide measurement data over a wireless network connection. Approach for enabling this vision has to be very different compared to the developments on the terminal equipment side. For cellular networks, WLANs, and other solutions trying, at least in some extent, to replace wired networks as the last hop solution, the

race has been towards higher bitrates and lower latencies, often at the cost of power consumption. This kind of approach is decidedly inappropriate for most sensor networking applications, as for battery-powered devices with long targeted deployment times, energy consumption is obviously a key issue. Additionally, in many machine-to-machine communication solutions and autonomous networking applications neither of the communication endpoints is usually that sensitive to jitter or increases in transmit times, allowing more aggressive sleep cycles to be used (naturally there are still application areas, such as industrial automation, where these broad statements are not appropriate; we shall deal with them separately in what follows).

Naturally, these considerations are not limited to sensor networks per se. In home networking, personal area networking, body area networking and in vehicular area networking similar considerations are valid in the more general context of embedded networks, or EmNets, as they are often called [3]. However, in the interest of clarity, we shall continue to use the term sensor network in a broad sense, understanding that remote actuation, control, and other kinds of applications might be involved. We shall use the term “embedded networks” to cover the cases when slightly more powerful nodes are present in the network, leaving the term “sensor networks” for networks formed by the smallest, most resource-constrained devices.

In the following we try to give the reader an overview of the present status of sensor and embedded networking, mainly from the research point of view, but not completely neglecting the various expected use cases and already available products. We start by having a look at the use cases available solutions, especially from the hardware point of view. After this we discuss the communication issues at some length, trying to answer questions like “how do we find these sensors?”, “how do we access them?”, or “how is the data flowing in the network?”. As a part of this discussion we introduce an approach that seems to

be very appropriate one for communication occurring at a single-hop scale. Following this discussion on the present state-of-the-art, we devote some time for looking further, first by identifying some open research issues, and in the end by letting ourselves loose from the shackles of present-day technology, and trying to picture what embedded networking might look like in the long run.

USE CASES

A large number of possible use cases and scenarios have been presented for wireless sensor networks and EmNets. Originally the interest for military use has been strong. Sensor nodes could be deployed from (possibly autonomous) aircrafts over a large territory, and the activities in the said territory could be monitored via the sensor readings. Demonstrations of the viability of this approach have already been presented by University of California at Berkeley, together with industry partners (see, for example, [4]). Similar operating modes could be applicable also to various civilian emergency contexts. In this case sensor readings could, for example, be used to detect survivors in a catastrophe area. The main requirements for both of these use cases are the need for reliability, and communication of the sensor readings over large distances, leading most likely to the need for multihop communications. While the pricing concerns naturally exist, in general the targeted price for these applications can probably be higher than in the consumer and commercial contexts. Also, a good professional application area example also falling in this category is meteorology, where professional systems can afford quite sophisticated equipments in price terms, but the considerations for deployment times and power consumption might still be there, depending on the exact application.

In the everyday civilian use two broad classes of applications for sensor networks are foreseen. "High-end" sensors characterized by greater robustness and need for reliability will most likely be used, for example, in various industrial and building automation applications, in cars and other motor vehicles, and by the medical community. Cheaper, more "mass-production" types of sensors will finally be used in different consumer/end-user applications to provide simple remote sensing, remote control, and information broadcasting services, the lowest level being a kind of "throw away" sensor which are very cheap, but probably a high price is paid in robustness and longevity. More general embedded networks will supplement these applications by adding various networking capabilities to more powerful devices, such as consumer electronics, cars, motorcycles, and various terminals. In any case it can be stated that at present the existing EmNets and sensor networks are usually very application specific, and only few building blocks for generic application platforms exist in commercially deployable state. There is a number

of hard research problems to solve before generic sensor networks and EmNets can be commercially exploited in wide scale. We are also only starting to understand the spectrum of possible applications for these kinds of networks in the consumer setting, so even requirements gathering is still very much a work in progress. New business models are undoubtedly going to be created, and commercial opportunities for innovative products will certainly be plentiful.

OVERVIEW OF WHAT IS THERE

The Mote-series of sensor platforms (see, for example, [5]) developed by University of California at Berkeley and by Crossbow Inc. have become one of the best known research tools in sensor networking. They have also served as a basis for a substantial number of field trials of various sensor networking demonstrations and applications. A number of sensor platforms of similar style have been developed at other universities and research institutes. For embedded networks the standard present-day research platforms are various embedded PCs, PDAs, Smartphones, tablet PCs and laptops, all of which are offered by a number of manufacturers.

On the software side TinyOS [6], also developed at University of California at Berkeley has become during the last year the de facto standard operating system for extremely resource-constrained devices. Being component-based, the developer has a choice of building blocks from which to compose and compile an operating system image that is most well suited for the application. Most of the TinyOS is written in nesC [7], which is fundamentally an extension of the "C" language, but is specifically extended to support the component structure of the TinyOS applications, and the event-driven concurrency model of the operating system. Since a large part of the Berkeley tools are freely available they have made a great service for the research community, and in large part helped to speed up the trend towards EmNets. For more powerful devices the number of available operating systems is larger. Specialty real-time operating systems are being used in numerous time-critical applications. From the "generic" OS side, Windows CE, Symbian and Linux are most often found in EmNet testbeds.

NEAR FUTURE SENSORS

As the research in sensor-related technologies is being vigorously pursued, number of exiting developments are expected to take place in the coming years. Below are some probable trends in sensor hardware platform development. As we are not really experts in the physical layer issues nor in electronics, we shall be rather brief.

Disappearance. Sensor hardware is becoming smaller and smaller, in some cases approaching the "smart dust" ideal [2]. Several research groups have

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Protocol				Flags				Length							
Length(cont.)								Source Port							
Destination Port								Data							

Fig. 1. NanoUDP header.

already demonstrated cubic-millimeter class sensor nodes with basic communication and processing capabilities, and mass production is looming in the horizon. Taken to completion, this trend can conceivably lead to an explosion in the number of sensor nodes, truly making the deployment of all-encompassing sea of sensors possible.

Integration. Most of the sensor hardware in use today is still constructed on small PCBs using basically off-the-shelf components. However, chip manufacturers have had the capability for designing chips that contain the sensing, communication and processing parts, simplifying considerably the needed PCBs, and simultaneously helping in bringing down the power consumption of the sensor nodes. Greater integration is also a prerequisite for mass production of cheap sensor hardware. For most low power radios and sensors the existing CMOS-technology is advanced enough to provide capability to build small devices with very low power consumption. For more esoteric applications requiring extremely low power consumption, more recent processes, such as SiGe, provide excellent opportunities.

Communication. Low-power radio technologies are developing at great speeds, increasing the deployment times of the battery-operated wireless sensors. For more powerful sensor nodes the availability of the recently standardized IEEE 802.15.4 (Zigbee) -radio will make the communication with more powerful terminals easier, as the Zigbee wireless interfaces are a strong candidate for proliferation in wireless terminals in the near future. Also future Bluetooth evolutions, reducing the power consumption of the wireless interface could be very interesting, especially considering the wide acceptance of the current Bluetooth. Finally, in slightly more distant future UWB technologies might eventually become an interesting choice for certain sensor network applications.

Power. Low-power electronics design is progressing as a field, further bringing down the power consumption of the sensors. Combined with the gradual improvement in battery technologies, the expected deployment times of the sensor nodes are constantly becoming longer.

MOVING DATA AROUND

MEDIUM ACCESS

Before discussing network layer issues, it pays to study a moment what is happening in the MAC-

sublayer, as the choice of MAC solution can have a large impact on the performance and power consumption of sensors or embedded network nodes. If the amount of traffic in the sensor network is very small, simple ALOHA-type of MAC can perform quite well. However, unless some guarantees can be given about the density of the network nodes, or the amount of traffic in the network, more elaborate MAC protocol becomes a practical necessity. Solutions developed for powerful terminals, such as the IEEE 802.11 MAC used in WLANs, are not usually satisfactory when applied in the “deep embedded”-context, most often the main problem being excessive power consumption. In the WLAN case this is due to the high idle-state power consumption, and expensive carrier sensing technique used.

Naturally a number of sensor network MAC solutions have been suggested. The S-MAC [8] uses periodic sleep cycles together with clustering and radio sleeping techniques to reduce power consumption. While this happens at the cost of increased latencies, as discussed above, this is not usually critical in sensor networking context. Reduction of energy consumption by a factor of 2-6 is reported by the authors, compared to an 802.11-like MAC. The present TinyOS versions use by default the B-MAC which is a CSMA/CA-based MAC scheme with application level feedback and which exploits various Clear Channel Assessment techniques. At present it seems that the B-MAC offers a very solid basis for efficient communications in sensor networks. There is a number of different suggestions that are based to different idle mode and CSMA combinations, one example being nanoMAC [9], which is to be ported also to TinyOS in the near future.

For EmNets in general a large number of MAC techniques have been proposed, but we shall not attempt a review here due to the sheer volume of material. Instead we refer the reader to the recent survey [10]. An interesting, more theoretical approach to the capacity questions touched by the MAC design, as well as the higher layer issues, is presented in [11].

SINGLE-HOP APPROACH

Let us now move up the stack to consider data movement as perceived and handled by the network layer. If an architectural assumption can be made, limiting the communication between sensors or EmNet nodes solely within the single-hop distance, we can perform heavy optimizations in the network layer, compared to traditional multihop networks. We shall devote this section into discussing these alternative solutions, and discuss more traditional network and transport layer solutions (including TCP/IP stacks) from the EmNet/sensor viewpoint in the next section. We introduce nanoIP as a representative, simple example of these alternative protocols. However, we do not claim it to be the final word in the topic, and warmly encourage further experimentation with these types of protocol

architecture (of course, the reader is free to use nanoIP as a basis for further work if he/she so prefers).

In overwhelming majority of the link-layer technologies developed, each interface is given a MAC-address, which is usually at least locally unique. The purpose of introducing a network layer address space is to retain the identity of the originator in the packets transmitted, even when the packet is transmitted over multiple links. Naturally, if we limit the number of hops to one, this additional address space is no longer necessary (whereas the transport layer address space used for application multiplexing still serves a purpose, at least in more general-purpose sensor networks). The removal of the network layer addressing also trivially makes all the functionality related to mappings between these two address spaces unnecessary. In the Internet world this would refer to protocols such as ARP, RARP, parts of DHCP, and so on. Naturally, for protocols utilizing only MAC-addresses to work correctly, some care is required in forwarding techniques employed below link-layer. Simple-minded physical layer relaying is naturally non-problematic, as it basically can be viewed as an extension of the radio range. A "Layer 1.5" bridging, retaining the frame structure and the MAC addresses, but running a simple bridging algorithm to reduce the overhead due to unnecessary transmissions, would also work. However, classical bridging that would change the source-address in the MAC header used to identify the sending endpoint of a link is problematic to use. In [12] an example architecture and protocol design based on the observations outlined above was introduced. We shall now concisely review this "nanoIP"-framework.

The basis of our nanoIP-family of protocols consists of two network/transport-layer protocols, called the nanoUDP and nanoTCP. In the single-hop approach there is no longer an acute need to differentiate between network and transport layers, and we have usually chosen the convention that the network layer is considered to be empty, while transport layer contains all the functionality. Naturally, this division is more a conceptual one, really existing only in terminology. NanoUDP is the basic connectionless transport protocol, meant for general-purpose communication for applications that either do not require reliability, or choose to implement reliable communication in the application layer. Fig. 1 shows the five octets long header. The version-field is used to distinguish between nanoUDP, nanoTCP, and possible future versions of the protocols. The flags are used mainly for fragmentation, distinguishing between first, last and middle fragments of the packets. Length-field is naturally used to carry the length of the packet (this is necessary as many link-layer technologies either do not report the frame payload lengths correctly, or employ padding to ensure that the frame sizes are always above some specified limit) and, in the case of fragmented packets, doubles as a fragment offset pointer. The rest of the space is consumed by

the port numbers, which are used, as mentioned above, to differentiate between applications. Due to the inherent unreliability of wireless channels, most link-layer technologies already include rather strong checksums for checking the integrity of the transmitted frames. Because of this, it was felt that adding a checksum for the nanoIP is not really necessary. However, we have considered an evolution of the nanoIP transport protocols with a limited support for optional headers, one type of which would be used to carry optional checksums.

For reliable communication the nanoIP-stack offers the nanoTCP-protocol. As is shown in fig. 2, the beginning of the header is exactly as for nanoUDP, but sequence and acknowledgement numbers have been added. We chose to keep nanoTCP stream-oriented, byte-based protocol as its hugely successful Internet cousin is, but arguably some additional space could be saved by resorting to packet-based ARQ. Due to the single-hop nature of the nanoTCP, no congestion control mechanisms have been implemented (though we are still going

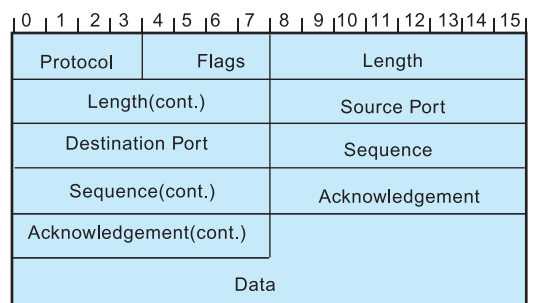


Fig. 2. NanoTCP Header.

to perform a number of experiments touching these issues, possibly leading into a further evolution of the protocol in the future). Window size is fixed for the duration of the connection, so no separate Window-field is necessary. For the determination of this fixed window size either a heuristic based on the link-layer characteristics can be used, or an optional header could again be defined. We are fully aware that especially with dense networks our approach has its own scalability "dangers", but nevertheless consider the present structure of nanoTCP as an interesting starting point for further studies.

While the optimization of basic networking protocols is extremely important if we are to arrive at successful sensor network protocol architecture, also the higher layer protocols should be considered. Many of the middleware protocols used in today's Internet are unacceptably heavy in terms of parsing complexity and the sheer size of the messages needed to enable most basic functionality. Both of these problems are often because of the text-based approach adopted in these protocols, often based on XML. In its present form the nanoIP-stack includes miniaturized and binary-based versions of two important middleware protocols, HTTP and SLP, called nanoHTTP and nanoSLP, respectively. The basic structure of the protocols remains in both cases

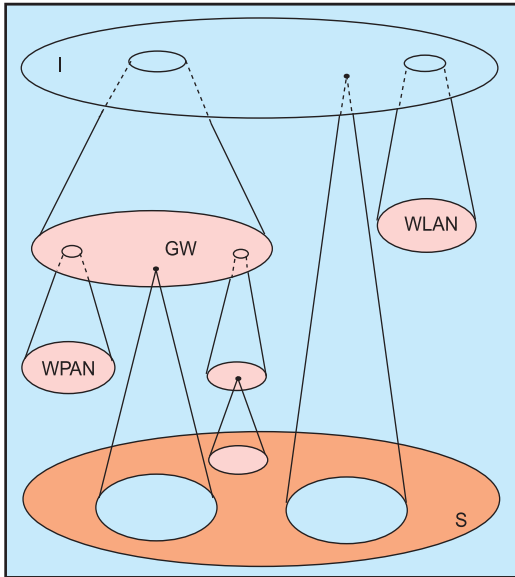


Fig. 3. The architecture of the future networks, featuring the Internet, sensor networks, and various EmNets and gateways between the two (the figure has been adapted from [1]).

untouched, making it easy to design gateways to convert between the Internet and nano-versions of the protocols. For compression of XML-style content, the WBXML-approach designed by the WAP-forum is quite effective.

Multihop Approaches

If the single-hop approach is not usable because of the application scenario, we have to decide on the network and transport layer techniques used to propagate the data across the multiple hops. The “traditional” solution is, of course, the use of some form of network layer routing, which attempts to deliver packets originating from a given node, to a given destination node. For more powerful embedded networks the adoption of TCP/IP stack is likely to be the approach of choice, given the level of acceptance of the current Internet protocols. As most embedded networks should work without an infrastructure, different ad hoc routing protocols are typically considered as being most relevant in this category. We shall not attempt a thorough review of this vast topic here, but instead refer the reader to the excellent accounts [13] and [14]. We would like, however, to point out that due to the very hard impact of multihop ad hoc routing to the network performance (see, for example, [15] and references cited therein), we expect that EmNets will not evolve towards massive multihop networks, and that the value of ad hoc routing protocols will lie mostly on enabling of zero-configuration networking. Most consumer EmNets will consist of one or two hops before a gateway connecting them to the infrastructure (see fig. 3 for illustration), but in some special applications there will most likely be the need for massive multihop networks, as mentioned earlier.

We should conceptually also distinguish between “vertical” and “horizontal” hops. With these “directions” we refer to fig. 3, meaning that

“vertical” hops might be necessary to connect some embedded network nodes to the infrastructure through a gateway (or possibly several) even though only one or two “horizontal” hops are used within a given embedded subnet. This will most likely to be the case with most PANs, BANs and VANs. The need for multiple “vertical” hops through gateways is due to the expected hierarchies these networks will form. In large sensor networks, on the other hand, it might be necessary to propagate data over a large number of hops. A number of alternative solutions to ad hoc routing have been proposed to accomplish this data propagation in an efficient manner, in many cases letting go of the specific identifiability of the communication endpoints, and adopting a data-centric approach. This will also lead to very heterogeneous networks of complex structure, as the sensor networks will very likely connect to PANs and other networks of “traditional” type.

We shall now shortly discuss some illustrative examples of the alternative data propagation solutions. In [16] two “SPIN” protocols improving classical flooding techniques were introduced. SPIN-1 is a simple 3-stage handshake protocol, which is augmented in SPIN-2 by energy conservation heuristics. The handshake procedure is used eliminate unnecessary data transmission occurring in classical flooding, and resource-awareness is used to adapt the behavior of the protocols according to energy resources. While the sensor network utilizing SPIN-type protocols is in a sense structureless, clustering has also often been suggested as a basis for energy efficient protocols for sensor networks. An interesting approach along these lines is LEACH [17], which is a clustered TDMA-based approach. Clusters are formed dynamically, and the cluster heads create schedules giving transmit times for other cluster sensors, allowing them to sleep rest of the time. Finally, directed diffusion [18] is an approach based on named data. The protocol sets up “interest gradients” for named data in the network following an initially broadcasted interest message, which can, for example, be flooded through the network. Network nodes then forward, or “diffuse” the named data as it becomes available along the interest gradients, simultaneously reinforcing the gradients on the propagation paths. Variants of the original directed diffusion concepts are being actively studied, and also practical experiences are being gathered using the readily available implementations for Berkeley MOTES and other platforms.

Regardless of the data propagation model adopted, it is very unlikely that wireless multihop sensor networks will be running standard TCP/IP stacks. The main reason for this is the high communication overhead, causing high energy consumption in the wireless interface. However, on some wired or fixed sensor networks, for which power consumption is not an issue, at least partial or interoperable implementations of the TCP/IP-protocols might become feasible. The innate

complexity of the TCP and IP is still manageable, even in the deep embedded environment. This is demonstrated by the existence of various small-footprint TCP/IP implementations (such as [19] amongst others), and also some hardware implementations, such as the WebChip design of the authors ([20] and [21]). Especially the hardware implementations show that the energy consumption and processing time associated with the parsing and storing the TCP/IP headers is completely negligible when compared to the energy and time consumed by the network interface.

Two design decisions for minimal TCP/IP implementations warrant further discussion. For complexity reasons several authors have chosen to implement only a subset of full TCP/IP functionality. While some of the functionality in full TCP/IP implementation seems too “obviously” be of no use in EmNets or high-end sensor networks, these kinds of decisions should always be made with great care. We still do not have sufficient understanding or expertise to proclaim that, say, congestion control will not be necessary inside embedded subnets or sensor networks (this is, of course, an issue also for other protocol stacks). While typical data sources operate rather infrequently in sensor networks, we still have to be mindful of the possibility that any given networking architecture will be employed to carry traffic with very different kinds of characteristics compared to the intuition of the designer. We might very well see first congestion collapses (especially if simple MACs are used) in larger sensor network testbeds in the near future. Other decision mentioned is related to the IP version used. Naturally the advocates of IPv6 see EmNets as a compelling reason for adopting it as the basic networking protocol. However, the proliferation of networks based on gateway architectures (employing network address translation techniques) may give the IPv4 address space surprising longevity.

Higher Layer Protocols

As mentioned before, middleware and application designs will have a large impact on the performance and usability of EmNets and sensor networks. Especially service location protocols need to be evolved, and various abstractions have to be developed. However, the field of middleware technologies is so large, that due to space constraints we are unable to make even an attempt of a review here that would make justice even to the most important technologies. Instead, we refer reader to the recent IEEE Network issue [22] and references therein, where these issues are discussed at length, and go on to discuss some topics that in our minds warrant a strong research effort in the future.

SOME OPEN RESEARCH ISSUES

MANAGEMENT OF SENSOR NETWORKS

Assume that we have deployed a sensor network of

some considerable size, and suppose further that after some time we want to make changes into the manner the network operates. At present we do not seem to have a good solution available. As the number of nodes in the sensor network (which can be quite loosely defined to begin with) can be potentially extremely large, it is hardly practical to try to identify the individual nodes and settings in their communication protocol stacks or in the applications. We need to develop techniques to efficiently manage ultra-large networks by efficiently aggregating vast amounts of configuration data, and having efficient heuristics for anomaly reporting and self-reparation. All of this should be accomplished in accordance to the very stringent limitations imposed by the simplicity of the hardware resources available in the sensor nodes. Probably this means that many of the functions mentioned above have to be implemented “above” the sensor sea, perhaps in an overlay formed by the gateway devices.

We should probably reconsider the authentication and authorization questions also against this network management background. These security and trust-related problems are hard enough for basic sensor data reading. If we desire to have multiple levels of authentication (to distinguish between, say, somebody allowed to just read the data, and somebody authorized to change the sensor network configuration), the problems become more difficult still.

Another can of worms is the question of software updates. While it might be argued that the software used in the sensors is so simple and easy to understand that it should be possible to make it flawless, this will hardly be the general case. Even the sensors available today are comparable in complexity to the computer systems used in the early space programs, and even those computers were occasionally programmed with flawed software, despite the quite substantial investment of manpower for the software development. Of course, it must be granted that today we have advanced compilers and programming environments at our disposal, but it is still quite easy to leave a flaw in the software that is extremely difficult to detect in automated manner. In any case software updates are of interest not only to correct flaws, but also to add new functionality, and for numerous other reasons.

In a simple case, such as a trivial offset in sensor calibration, it is most likely easiest to solve the problem at the gateway or end-user application level. For more substantial problems, the possibilities are at least the following:

1. Complete reprogramming of the sensor’s program memory, followed by a “reboot”.
2. Partial reprogramming replacing only the incorrect parts of the code.
3. Upload of a correctly working version of the code, followed by a configuration change.
4. Physical redeployment, replacing/accompanying the defect sensors with updated ones.

What is meant by the first two should be obvious. The third approach might be useful for sensors running their programs mainly from ROM, but with some reprogrammable memory containing some basic operating system or application configuration data. Of course, suitable design of the operating system or application is a necessary prerequisite for all of the first three. The main concern, however, is not how to implement this functionality. This can, and, indeed, has been already done (see, for example, [23]). The problem is to make a secure implementation, that would not allow simple reprogramming with malicious code (note that aggressive re-reprogramming might not help, especially if the malicious code can prevent further updates). Of course there might be applications where this is not really a concern (sensors that are in a physically secure location being the prime example), but in most environmental, building monitoring, emergency use, military use, and medical use scenarios great damage could be done by suitable malicious code. The security concern is especially acute if the reprogramming is done by “viral” or “epidemic” code, that is, as a software update that makes copies of itself, and transmits these copies into the neighboring sensors, which then repeat the process. This approach has been suggested numerous times in the sensor networks context, see, for example, [24], [25] and [26]. If secure reprogramming turns out to be an unfeasible solution, we are forced either abandoning or redeploying the sensor network. These options might also turn out to be surprisingly difficult to follow through. In the case of various public networks there might be a need to convey to the general public about the unreliability of the information supplied. The sensors might also be difficult to remove, as might be the case in various building automation uses (sensors inside walls monitoring the structural integrity being prime examples). In any case, redeployment incurs additional costs, which might be considerable, despite the very low price of a single node.

Context and Cognition Issues

An often presented potential application for sensor networks is the creation of context sensitive applications. A relatively simple, and an often cited example would be a terminal, such as a cellular phone, adapting its user profile to the changes in the ambient conditions, such as movement profile, lightning conditions and the noise level. An interesting account on context sensitive used interfaces is [27]. Slightly more complex example would be adaptation of an intelligent house into the changes of conditions outside and inside, such as changes of temperature, sunlight levels, and movements and gestures of the residents. All of these applications require the automated making of somehow “intelligent” decisions from a vast flood of data from a possibly very large number of sources.

Most of the research on context sensitivity related to the above applications has been focusing on the classification of sensor data into a small number of

classes, and then applying some decision-making algorithm for sensor readings, seeking to match one of the classes with a high confidence. Even this approach has turned out to be surprisingly difficult to pursue with any degree of reliability. Much more complex is the problem of doing “generic” deductions from the data that cannot be anticipated in the original development time. As pointed out in [28], most likely some kind of hard artificial intelligence turns out to be necessary, something that has been evading AI-researchers for decades, despite of monumental effort and high-quality research. In any case, this just shows how wide-ranging problems have to be tackled before more futuristic sensor networking applications can actually be realized.

Topology of Sensor and Embedded Networks

During the past few years there has been a small revolution starting and going on in understanding the topological structure of the Internet (see, for example, the review articles [29] and [30] for a full account), and new topological discoveries are still being made (a recent interesting example being [31]). So far, very little is known of the topologies of future sensor networks, and of different embedded networks. Because of the way these networks will be constructed, most of the results regarding the topologies of fixed networks will no longer be valid. Also, due to the nature of the network nodes, the network topologies will often be extremely dynamic, and stochastically determined. As we already know, the topology of the network can have a profound impact on diverse phenomena such as resilience against node failures, search algorithm convergence speeds, and transport layer protocol efficiencies. Against this background there is definitely a need to develop a more complete understanding of the embedded network topologies, and study the ways these topologies can be influenced, if certain specific properties turn out to be particularly desirable. It may turn out that tools of stochastic geometry and statistical physics will become everyday companions to those of graph theory in analyzing these massive networks.

VISIONS OF THE FUTURE

If we are able to resolve the immediate research problems outlined in the previous sections, and sensor and embedded networks truly become pervasive, and natural parts of our everyday lives, we might start facing very different problems to study. From the pure networking point of view the complexity of the “Internet”, or whatever the conglomerate of all the networks will be called in the future will become truly extraordinary. As the number of networked nodes starts to approach 10^{15} , we start to move to the levels of complexity normally found in systems studied by biologists and statistical physicists. Extremely well-scalable information processing techniques have to be developed, unless we choose to let go of the hope of maintaining even remotely large-scale picture of the networks.

Ethical and privacy questions might also become topical if sensor networks become truly pervasive in our surroundings. Our guess is that many would feel slightly uncomfortable living a life surrounded by “ambient intelligence” formed of sensors and effective cognitive techniques, making it possible to actually make meaningful inferences of high abstraction levels from the sensor data. Legal steps might become necessary to prevent the emergence of a “big brother” society. Other danger can be called the “little brother society”, namely that ubiquitous sensors could also facilitate voyeurism and media misuse virtually by everyone. These considerations might become especially acute if sensor implants, and body-embedded networks ever become a reality (and so far there are no reasons to think that they will not).

On a lighter note we should also be aware of the impact of various society-wide trends might have on embedded networking. Environmental concerns might create a call for what we call “GreenChips”, or environmentally friendly sensors. Some developments in this direction are already taking place as major chip manufacturers have announced reductions in the lead-content of the produced microchips. Similarly there could be a thrust for further development of power sources based on different techniques available for harvesting energy from our surroundings.

CONCLUSIONS

Sensor networks and EmNets are one of the most active research areas in the networking field, and have a potential for truly disruptive research results. While the advances in these fields have been great, there is no sign of the field beginning to wither, or go into decline. On the contrary, as the basic problems related to fundamental enabling technologies are being understood, we start facing extremely difficult problems in the higher layers of abstraction. It is quite likely that in a relatively short future new research challenges have emerged that we are still blissfully ignorant of. This is what makes EmNets so exciting.

ACKNOWLEDGEMENTS

We thank financial support from DFG, Microsoft (Microsoft Research and European Microsoft Innovation Center), the Academy of Finland (grant 50624), Ericsson Research, and European Union. We would also like to thank our colleagues in the nanoIP team, especially Zach Shelby. The nanoIP research has been carried out in collaboration with the Center for Wireless Communication, University of Oulu, Finland.

REFERENCES

- [1] P. Mähönen, J. Riihijärvi, M. Petrova, and Z. Shelby, "Hop-by-Hop Toward Future Mobile Broadband IP,"

- IEEE Communications Magazine, March 2004, pp. 138-146.
- [2] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next Century Challenges: Mobile Networking For 'Smart Dust'," Proceedings of the MOBICOM, 1999, Seattle, pp. 271-278.
- [3] "Embedded Everywhere," National Academy Press, Washington D.C., 2001.
- [4] <http://robotics.eecs.berkeley.edu/~pister/29Palms0103/>
- [5] J. Hill and D. Culler, "Mica: A Wireless Platform for Deeply Embedded Networks," IEEE Micro., vol 22(6), Nov/Dec. 2002, pp. 12-24.
- [6] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System Architecture Directions For Networked Sensors," ASPLOS 2000.
- [7] David Gay, Phil Levis, Rob von Behren, Matt Welsh, Eric Brewer, and David Culler, "The nesC Language: A Holistic Approach to Networked Embedded Systems," Proceedings of Programming Language Design and Implementation (PLDI) 2003, Jun. 2003.
- [8] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," Proceedings of the IEEE Infocom, New York, NY, USA, Jun. 2002, pp. 1567-1576.
- [9] J. Haapola, M.Sc. thesis, University of Oulu (unpublished).
- [10] R. Jurdak, C. V. Lopes, and P. Baldi, "A Survey, Classification and Comparative Analysis of Medium Access Control Protocols for Ad Hoc Networks," IEEE Communications Surveys, First Quarter 2004, vol. 6, no. 1.
- [11] P. Gupta and P. R. Kumar, "The Capacity of Wireless Networks," IEEE Transactions on Information Theory, vol. IT-46, no. 2, pp. 388-404, Mar. 2000.
- [12] Z. Shelby, P. Mähönen, J. Riihijärvi, O. Raivio and P. Huuskonen: NanoIP: The Zen of Embedded Networking, Proceedings of the ICC 2003, May 2003.
- [13] C. E. Perkins, "Ad Hoc Networking," Addison Wesley Professional, 2001.
- [14] C.-K. Toh, "Ad Hoc Mobile Wireless Networks: Protocols and Systems," Prentice Hall, New York, 2002.
- [15] G. Holland and N. H. Vaidya, "Analysis of TCP Performance Over Mobile Ad Hoc Networks," Proceedings of IEEE/ACM MOBICOM '99, August 1999, pp. 219-230.
- [16] J. Kulik, W. Rabiner, and H. Balakrishnan "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," Proceedings of MOBICOM, 1999, Seattle, pp. 174-185.
- [17] W. R. Heinzelman, A. Sinha, A. Wang, and A. P. Chandrakasan, "Energy-scalable Algorithms and Protocols for Wireless Microsensor Networks," Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP), Jun. 2000.
- [18] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCOM '00), Aug. 2000, Boston, Massachusetts.

-
- [19] A. Dunkels, "Full TCP/IP for 8-Bit Architectures," Proceedings of the First International Conference on Mobile Systems, Applications, and Services (MobiSys03), San Francisco, May 2003.
- [20] J. Riihijärvi, P. Mähönen, M. Saaranen, J. Roivainen, and J.-P. Soininen, "Providing Network Connectivity for Small Appliances: Functionally Minimized Embedded Web-Server," IEEE Communications Magazine, vol. 39, no. 10, pp. 74-79, Oct. 2001.
- [21] J. Roivainen, J. Riihijärvi, P. Mähönen, J.-P. Soininen, M. Saaranen, "Minimisation of Functionality and Implementation of Embedded Low-Power www-server," IEEE Electronics Letters, vol. 38, no. 2, pp. 100 - 101, 2002.
- [22] IEEE Network, vol. 18, no. 1, January/February 2004 and articles therein.
- [23] J. Lifton, D. Seetharam, M. Broxton and J. Paradiso, "Pushpin Computing System Overview: a Platform for Distributed, Embedded, Ubiquitous Sensor Networks," Proceedings of the Pervasive Computing Conference, 2002.
- [24] P. Levis and D. Culler, "A Case for Infectious Virtual Programs in Sensor Networks," unpublished.
- [25] P. Levis and D. Culler, "Mate: A Tiny Virtual Machine for Sensor Networks," Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems, San Jose, CA, USA, Oct. 2002.
- [26] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks," Proceedings of the First Symposium on Network Systems Design and Implementation, NSDI '04, Mar. 2004.
- [27] B. L. Harrison, K. P. Fishkin, A. Gujar, C. Mochon, and R. Want, "Squeeze Me, Hold Me, Tilt Me! An Exploration of Manipulative User Interfaces," Proceedings of the SIGCHI conference on Human factors in computing systems, Los Angeles, pp. 17 - 24, 1998.
- [28] T. Erickson, "Some Problems with the Notion of Context-Aware Computing," ACM Communications, vol. 45, Iss. 2, pp. 102-104, Feb. 2002.
- [29] R. Albert and A.-L. Barabási, "Statistical Mechanics of Complex Networks," Reviews of Modern Physics, vol. 74, no. 47, 2002.
- [30] M. E. J. Newman, "The Structure and Function of Complex Networks," SIAM Review 45, pp. 167-256, 2003.
- [31] S. Zhou and R. J. Mondragon, "The Rich-club Phenomenon in The Internet Topology," IEEE Communication Letters, vol. 8, no. 3, pp. 180-182, 2004.