CONET Newsletter

INFSO-ICT-224053 www.cooperating-objects.eu Issue nr.14, July 2011

Contents

- Editorial
- Quo vadis, WSN Internet Integration?
- SPINE (Signal Processing In Node Environment) Framework
- Member Profile: Distributed Systems Group, ETH Zurich
- Announcements

Editorial

Welcome to the 14th issue of the CONET newsletter. CONET is the EU FP7 network of excellence on Cooperating Objects, merging the fields of embedded systems for robotics and control, pervasive computing and wireless sensor networks. CONET focuses on establishing the field of Cooperating Objects within the research and industrial community, thus strengthening the position of Europe in the research landscape.

This issue provides an industry viewpoint on cooperating objects: a guest column by Coalesenses GmbH – a spin-off from the University of Lübeck, Germany – discussing the integration of sensor networks into the Internet; and an article by CONET member Telecom Italia on their well-known SPINE framework for signal processing in body sensor networks. The member profile of this issue is devoted to the Distributed Systems Group at ETH Zurich, Switzerland.

This issue has a strong flavour towards communication technologies for cooperating objects as well as on the emerging domain of smart grids. We also report on upcoming events including the yearly organized CONET Summer school.

If you are interested in obtaining up-to-date information about the CONET project please visit our website at: http://www.cooperating-objects.eu

We hope you will enjoy this issue!

■

Quo vadis, WSN Internet Integration?

By Carsten Buschmann, coalesenses GmbH

Over the last years, wireless sensor networks (WSNs) have attracted a considerable amount of research effort. However, the focus changed over time. In the early years researchers aimed at low level aspects like getting sensor network running despite of resource limitations, increasing the power efficiency of protocols and inventing new algorithms for obtaining context information. Contemporary work is targeted more and more at higher level aspects such as programming paradigms, interoperation and the Internet of Things.

In the past, the protocol stack used within sensor networks and in the Internet differed significantly. A gateway node connected the sensor network to a wide area network and converted data on the application level. While this approach serves the goal of attaching sensor networks to the internet, it does not really support integration. Whenever new functionality is added to the sensor network (e.g. by adding new nodes that support new sensors), the gateway also needs to be modified.

Internet Integration using 6loWPAN, IPv6 and CoAP

Consequently, true Internet integration must go further: rather than attaching networks, the Internet has to be extended into wireless sensor networks. Applying Internet technologies to sensor networks does not only promote interoperation and ease installation, but also leverages the availability of a wealth of tools, techniques and expertise.

To integrate sensor networks into the internet, a number of RFCs and internet drafts has been proposed: 6loWPAN [1] and its extensions describe an adaptation layer between IEEE 802.15.4 and IPv6 that cares for header compression, packet fragmentation and neighbor discovery. RPL [2] proposes a routing protocol that aims at covering the requirements of many foreseen WSN applications such as Building Automation, Home Automation, Industrial and Urban Networking. For the DYMO routing protocol [3], a mesh-under-networking variant called DymoLOW [4] has been proposed.

As services are a widespread paradigm in the Internet, there is also a demand for offering RESTful web services [5] in WSNs. However, the commonly used HTTP fails to meet WSN requirements due to its high complexity and overhead. Consequently, the Constrained Application Protocol (CoAP) [6] was developed with the demands of resource-constrained devices in mind: it offers similar primitives as HTTP, but operates via UDP rather than TCP, adds end-to-end reliability on the application layer, uses binary headers for increased efficiency, and provides a simple mechanism for the discovery of CoAP resources available on a node. With Observe [7], a powerful event based subscription and notification service was added.

We implemented an IPv4 and IPv6 dual network stack which comprises all functionality required for integrating wireless sensor networks with existing Internet and LAN installations using the Internet protocol family (see Figure 1).

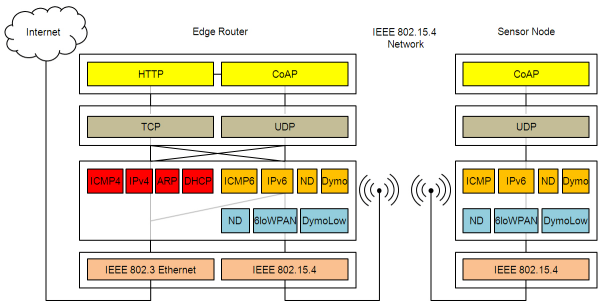

Figure 1: Protocols in the coalesenses IP stack

Within the sensor network, the 6loWPAN protocol suite (including implementations of neighbor discovery, header compression and fragmentation) is used to transmit IPv6 datagrams over the IEEE 802.15.4 link layer radio interface. The stack supports both the route-over mode with DYMO as a routing protocol and mesh-under mode with DymoLow for routing in the sensor network, functionality for routers within the network as well as for 6loWPAN border routers is included.

The two charts in Figure 2 show average ping round trip times for route-over and mesh-under configurations with different payload sizes and wireless hop counts. The reason for the large increase at the first wireless hop when sending larger payloads in the mesh-under configuration is that inter-packet intervals are stretched to 50ms if IPv6 datagrams are fragmented into multiple IEEE 802.15.4 packets. Like this, interference between subsequent packets on multi-hop paths can be prevented, but consequently the delay is increased.

Besides UDP and TCP, a simple HTTP server is part of the stack. In addition, it provides full-featured CoAP server and client implementation including Observe. Hence, the stack is ready to offer RESTful web services within wireless sensor networks as well as to the outside world. We provide a border router and a wireless sensor node for public experimentation [8], for querying CoAP services from a PC a Mozilla Firefox plug-in is available [9].

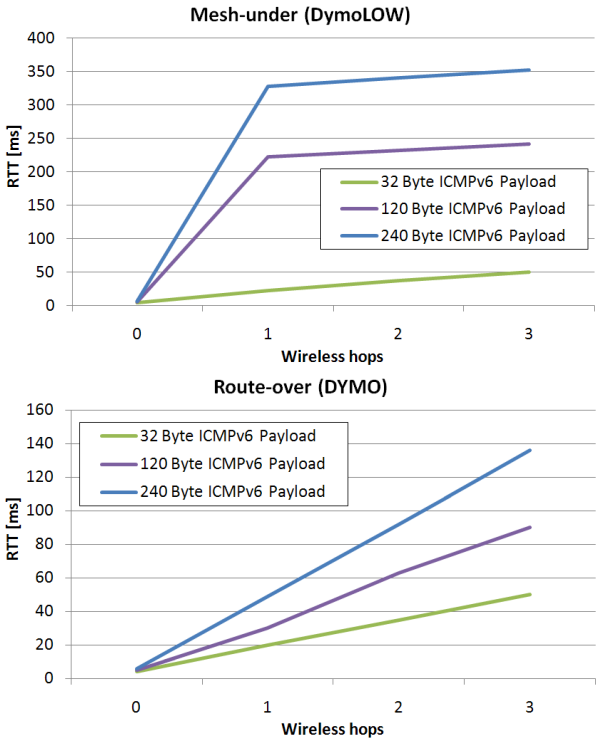

Figure 2: Ping times for mesh-under and route-over configurations

Research Challenges

However, with this framework at hand, still a number of challenges remain to make the vision of the omnipresent Internet of Things come true:

- while proprietary protocols allow for application-specific optimization of energy optimization, trade-offs have to be made when using 6loWPAN,
- time synchronization with a precision of a millisecond or better is not yet available,
- no clear security architecture has been settled yet,

- due to the many configuration options, true interoperation is still rather a promise than reality.

While the above list is not complete, two additional aspects should be pointed out.

One key question is how all the data from the Internet of Things can be interpreted appropriately to make it meaningful information and to enable developers to leverage it for new applications. Together with other European research partners, we approach this issue in the Spitfire [10] project by integrating Semantic Web technologies with wireless sensor networks. We propose to use CoAP services to retrieve machine-readable RDF descriptions [11] from so-called Semantic Entities (SEs), which can range from single sensors to complex structures like building automation networks. Like this, sensor and context information can be seamlessly linked into clouds of linked open data, which can be searched with powerful query languages such as SPARQL [12].

Another interesting issue is how WSN application development will change with new paradigms such as services being introduced. With CoAP, writing applications basically consists of implementing resources that either directly provide services or that call other services to collect data. This approach helps to separate functionalities into logical blocks and promotes component reuse. It also helps the developer to keep away from networking aspects, easing application development for programmers that are not WSN experts. On top, this development gives rise to new application development techniques based upon service orchestration. First steps into this direction are currently being done [13], and many other interesting publications will come up.

Contact information

Further information can be found on the website www.coalesenses.com. If you have any questions, ideas, or suggestions you can reach me by email at buschmann@coalesenses.com.

References

[1] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, “Transmission of IPv6 Packets over IEEE 802.15.4 Networks,” RFC 4944, http://www.ietf.org/rfc/rfc4944.txt

[2] T. Winter et. al, “RPL: IPv6 Routing Protocol for Low power and Lossy Networks“,http://tools.ietf.org/html/draft-ietf-roll-rpl-19, March 2011

[3] I., Chakeres, D. Perkins, "Dynamic MANET On-demand (DYMO) Routing", http://tools.ietf.org/html/draft-ietf-manet-dymo-21, May 2007.

[4] K. Kim et al, “Dynamic MANET On-demand for 6LoWPAN (DYMO-low) Routing”, http://tools.ietf.org/html/draft-montenegro-6lowpan-dymo-low-routing-03, June 2007

[5] R. Fielding, “Architectural styles and the design of network-based software architectures,” Ph.D. dissertation, University of California, Irvine, 2000.

[6] Z. Shelby, B. Frank, and D. Sturek, “Constrained application protocol (CoAP),” http://www.ietf.org/id/draft-ietf-core-coap-06.txt, May 2011

[7] K. Hartke, Z. Shelby, “Observing Resources in CoAP”, http://www.ietf.org/id/draft-ietf-core-observe-02.txt, March 2011

[8] Border router: coap://[2001:638:70a:b069::ff:fe00:2304]:23048, sensor node: coap://[2001:638:70a:b069:215:8d00:11:a400]:23048/

[9] Copper CoAP Plugin for Firefox: https://addons.mozilla.org/en-US/firefox/addon/copper-270430/versions/

[10] “SPITFIRE - Semantic Service Provisioning for the Internet of Things using Future Internet Research by Experimentation,” http://www.spitfire-project.eu/.

[11] W3C RDF Data Access Working Group members, “SPARQL Query Language for RDF,” http://www.w3.org/TR/rdf-sparql-query/.

[12] RDF Working Group, “Resource Description Framework (RDF),” 2004, http://www.w3.org/RDF/.

[13] N. Glombitza et al, “Using State Machines for a Model Driven Development of Web Service-Based Sensor Network Applications, 32nd ACM/IEEE International Conference on Software Engineering, 2010

■

SPINE (Signal Processing In Node Environment) Framework

By Roberta Giannantonio, Telecom Italia

The design of systems based on Body Sensor Networks (BSNs) is complex, particularly due to the challenge of implementing signal processing intensive algorithms for data interpretation on wireless nodes that are very resource limited and have to meet hard requirements in terms of wea-

rability and battery duration. Furthermore, debugging software on sensor nodes is very difficult and time consuming due to the lack of support architecture in embedded operating systems; redeploying the debugged code on the actual sensing devices takes significant amount of time as well. New abstractions and tools are needed to support application developers during design space exploration and fast prototyping.

SPINE (Signal Processing in Node Environment) is a software framework for the design of signal processing intensive WSNs, with special interest for wearable sensors and BSNs (Body Sensor Networks).

The main goal of SPINE is to provide developers with support for rapid prototyping of signal processing applications.

In SPINE, sensors and common processing blocks, such as math aggregators and threshold-based alarms, can be configured independently and connected together arbitrarily at run-time based on external controls. Such an approach allows heterogeneous applications to be built atop the same basic software components, enhancing code reusability and, more importantly, removes the need for redeploying the node side code based on a particular application.

Figure 1: SPINE Logo

SPINE is available as Open Source project under LGPL 1.2 license at http://spine.tilab.com. Current version of the framework is 1.3.

SPINE is composed of two different sides: node and server side. The node side is TinyOS2.x firmware and provides on node services such as sensor data sampling and storage, data processing and more. It provides interfaces to easily add new sensors and new functionalities. The SPINE server side provides local and remote applications with lightweight Java APIs to be used for managing the sensor nodes or issue service requests. The APIs are easily portable to devices of various capabilities, such as PCs or mobile phones.

The SPINE framework helps designers to evaluate the efficiency of distributed implementations of data classification algorithms with respect to the use of energy and channel bandwidth.

One of the key advantages of SPINE is the ability to satisfy diverse application needs at run-time, avoiding, in most situations, the costly redeployment of the code running on the remote sensing devices. In fact, processing functionalities that are very common in the WBSN domain, such as time-domain feature computation, threshold-based alarms, but also simple raw data transmission, are available by default during the first deployment of the node-side software. Furthermore, available sensors can be set-up, activated and disabled dynamically and their output can be arbitrarily connected to the on-line processing, adding a further level of flexibility.

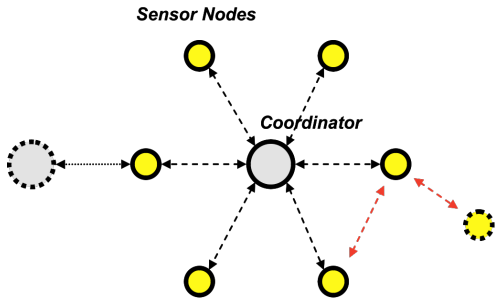

Figure 2: SPINE Network Architecture

Use of the Spine Framework

Efficient implementation of WSN applications requires appropriate allocation of the limited resources on the nodes in terms of power, memory and processing. This is especially critical in signal processing systems, which usually have large amounts of data to process and transmit. SPINE provides a flexible framework to support developers in evaluating the implementation tradeoffs along the memory, computation and energy dimensions. In particular, it supports both centralized and distributed implementations and offers developers tools to evaluate and select the architecture that is most suitable for the target application.

Centralized architectures maximize the amount of functions executed on the coordinator, which is usually implemented on a gateway or a mobile phone and therefore has more resources than sensor nodes. In a fully centralized approach, sensor nodes typically transmit the raw sensor data to the coordinator, which extracts features and executes classification algorithms. This implementation has the disadvantage of using large amounts of power and channel bandwidth for data transmission.

In distributed architectures sensor nodes share the burden of performing some classification functions. For example, sensor nodes may compute

features locally and transmit them to the coordinator. Sending only the result of a computation instead of transmitting the raw sensor data might significantly reduce the amount of transmitted data and therefore allow a more efficient utilization of the wireless bandwidth and relevant savings of the energy of the nodes. In some cases further resource optimization could be achieved by implementing some classification functions on the node.

Typically, distributed implementations require more effort on the part of the developer. However, SPINE reduces this effort significantly by providing readily available features on the node. WSN applications must be designed as dynamic systems that adapt to varying interactions with the environment and can be easily extended with new features during the lifetime of the system.

Data processing requirements may vary significantly during system operation, for example when the person being monitored performs different activities. In this case the middleware should dynamically adapt and quickly configure internal services and communication accordingly. The Service Layer of SPINE supports dynamic adaptation of the data collection and interpretation services by allowing the coordinator node to send messages to the sensor nodes specifying parameters such as the sampling rate and the features that sensor nodes must compute.

WSN application developers can use the built-in features of the framework such as radio controller, the feature library and engine, or can easily add new features and services taking advantage of the SPINE APIs and management functions. For example, support for new sensors can be easily added by writing the sensor drivers so that they comply with the generic sensor interface and without having to modify the sensor controller, the buffer manager or the feature engine. This also results in the sensor being made immediately available for further processing by other parts of the SPINE framework.

The implementation of the sensor node component of SPINE has been designed to decouple sensing, data processing and communication functionality. As a result, it is relatively easy to add new functionality and new sensor drivers. New features can be easily introduced as far as they conform to a defined feature interface. It is also possible to introduce new services or processing functions. Both the Feature Engine and the Alarm Engine conform to a defined function interface, which specifies that a processing component will be setup with one message from the base station, after which sub-functions may be each activated with an additional message. The format of setup messages and messages sent back to the base station is not defined by the interface, which gives the developer full freedom when developing new processing components. Functions and sub-functions are each addressed by a unique 8-bit value in the setup message, which allows for ample expansion of the SPINE Framework by the open source community.

SPINE 1.3 Data collection and node emulator tools

Designers of algorithms and application developers that want to test an implementation on data collected from the sensors sometimes have to cope with the lack of the specific hardware. To avoid this issue we developed the SPINE node emulator application that virtualizes the real hardware. In this way users with the hardware can collect, tag, store, and share data with other users. Moreover, the same sensor node might be used several times for different data acquisitions. Stored data can be used for testing different algorithms as if the nodes were real. This approach will easily allow comparing different solutions to the same problem (e.g. different algorithms solving the same classification problem), since the same data can be used several times. Furthermore, this will speed up the development process in a real application, since the application developed for virtual sensors will perfectly run with real sensors without the need of writing new code. For this propose we realized two different tools, namely SPINE Data Collector and SPINE Node Emulator.

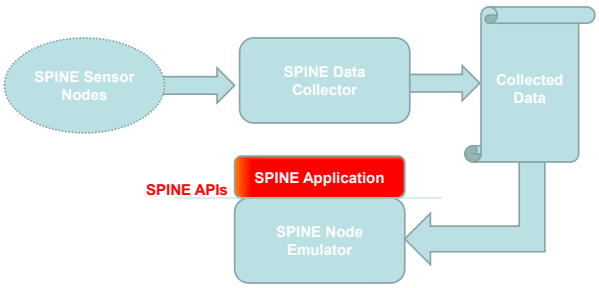

Figure 3: SPINE Data Collector and Node Emulator logical links

The main goal of the SPINE Data Collector application is to collect sensor data and store them in a common format. Therefore, Data Collector main functionalities are nodes configuration, sensor data collection and storage.

SPINE Node Emulator is a Java stand-alone application that uses collected data (Data Collector output) to emulate SPINE sensor nodes.

The same Java code that is programmed using the SPINE emulated data will run with real sensors without any modification. There are several advantages of using a SPINE emulator. For instance, processing functionalities can be implemented in the emulated environment first to simplify the debug process. Furthermore, data set from real sensors can be used to validate and compare objectively different processing algorithms or hardware set-ups. Finally, the emulator with a standard data set can be used by interested developers to investigate the potential of the SPINE framework itself, even if they do not have suitable physical sensor nodes.

Contact

Further information and all contacts can be found on the SPINE website http://spine.tilab.com.

■

Member Profile: Distributed Systems Group, ETH Zurich

By Silvia Santini, Kay Römer, Friedemann Mattern

The **Distributed Systems Group** (DSG) at ETH Zurich pursues research in the areas of Distributed Computing and Ubiquitous Computing.

The overall research activities of the DSG span the broad fields of models and concepts for distributed computations, ubiquitous computing, Internet of Things, sensor networks, infrastructure support for smart real-world objects, interaction in smart environments, and privacy and security concepts in the area of ubiquitous computing. In the following, we provide a brief overview of the main research activities of the DSG, as well as a short portrait of Mirasense and Koubachi, two spin-off companies that commercialize ideas and products that were first prototyped in the context of DSG research projects.

Wireless Sensor Networks

Enabling the practical design and deployment of wireless sensor network presents a number of challenging research problems. Since 2001, we are investigating these issues within the framework of NCCR MICS, the Swiss National Center of Competence in Research on Mobile Information and Communication Systems (www.mics.org).

In an early phase of MICS, we jointly developed the *BTnode system* together with other research groups at ETH Zurich. This system is a hardware and software platform for sensor networks that formed an important experimental environment for validating and evaluating our research.

We have also contributed to the definition of *novel programming paradigms* for sensor networks. In particular, we have been devising a declarative specification technique, called *generic role assignment* that allows programming a sensor network as a whole by automatically assigning roles to sensor nodes.

Further, we have investigates concepts and tools for facilitating the deployment of large-scale sensor networks for realistic applications. In practice, sensor networks are heavily affected by their unpredictable environment, such that deployments often fail despite extensive tests in laboratory settings. To cope with this problem, we developed tools to monitor deployed sensor networks by passively listening to network traffic and to detect failures using distributed assertions.

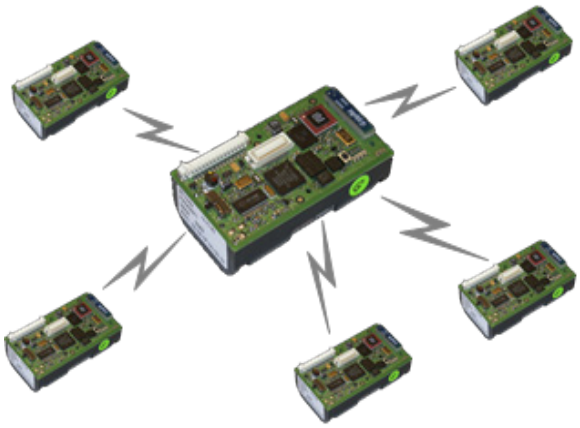

Figure 1: BTnode: a hardware and software platform for sensor networks.

Currently, we are investigating the systematic design of programming abstractions that allow runtime testing and debugging at the same conceptual level as programming. Recent work also includes the design of practical methods to characterize the influence of the topology on the performance of wireless sensor network communication protocols as well as the development of novel collection protocols that can achieve very high network lifetimes thanks to their adaptive behavior. Further, we are working on the design and implementation of lightweight active monitoring techniques to detect the presence and type of failures (e.g., routing loops) that may occur during the execution of collection protocols.

Internet of Things

The Internet of Things represents a vision in which the Internet extends into the real world embracing everyday objects. Physical items are no longer disconnected from the virtual world, but can be controlled remotely and can act as physical access points to Internet services. The contribution of the DSG to bring this vision closer to reality spans several different topics.

In cooperation with leading telecommunication companies, we explored the use of mobile phones as a building block for the Internet of Things. In particular, mobile phones can serve as ubiquitous gateways to smart objects and sensor networks embedded in the environment. In this context, we used mobile devices as people-centric sensors and as a gateway to sensor networks embedded in the environment, thus gaining online and real-time access to the state of real-world objects and environments. Building on this infrastructure, we also investigates approaches to enable real-world search for objects that exhibit a certain state at the time of the query – a challenging problem due to the scale and dynamics of the information space being searched.

More recently, we have been investigating how standard Web protocols and tools can be used on resource-constrained devices to enable a seamless integration of everyday items into the Internet. To this end, we explored the design of lightweight, resource-oriented architectures based on the REST (REpresentational State Transfer) paradigm and validated our work building several prototypes. Our work demonstrated how it is possible to manage the sheer number and heterogeneity of resource-constrained devices without the need of defining and implementing dedicated software and proprietary interfaces. While our previous work mainly focused on sensing devices, we are currently also investigating interaction paradigms that allow controlling actuators using standard Web protocols. Very recently, we have also developed first prototypes to connect things to the Web using programmable, low-power WiFi modules.

In the context of a recently acquired grant of the Swiss National Science Foundation (SNF, www.snf.ch), we are also developing core components that will set the stage for a transformation of the Internet of Things from a vague concept and a basic infrastructure to an open and accessible system integrated into the World Wide Web. By designing a scalable infrastructure, we aim at creating the architectural backbone for an environment of Web-enabled devices. We are also exploring possibilities of facilitating the Web integration and composition of real-world data and functionality while supporting people to actively participate and benefit from the Internet of Things.

Ubiquitous Computing

Ubiquitous computing technologies will have a strong impact on future society. In particular, *security and privacy* will be of prime concern in a world of highly interconnected, invisible devices that will eventually permeate our everyday lives. We thus explored privacy awareness concepts targeted at ubiquitous computing environments that allow data collectors to both announce and implement data usage policies.

A further research focus lies on exploring novel forms of interaction to allow humans to interact with smart objects and environments. For instance, we investigated the use of camera-equipped mobile phones for robust scanning of bar codes, addressing the challenges presented by realistic environments such as imperfect illumination and blurred images. This approach to object identification enables linking products with the abundance of information available on the Web so that consumers can quickly retrieve information and access services related to the scanned product. Mirasense (www.mirasense.com), one of the companies the DSG spun off in 2009, leverages this technology to offer novel product-related services to millions of mobile phone users.

Koubachi (www.koubachi.com), a further spin-off company of the DSG, also built its business case on facilitating interaction with everyday objects. In particular, Koubachi offers a system to monitor the health status of indoor plants and communicate the needs of the plants in an emotionally appealing way using their owner's mobile devices.

Further research on novel interaction paradigms included the exploration of *augmented toy environments*, where traditional toys such as a knight's castle are augmented with sensors and actuators to enrich children's play by sound effects, verbal commentary, and visual feedback. This also allows us to integrate an *interactive learning experience* into play. Beyond the design and implementation of such augmented toy environments, we performed user studies in cooperation with social scientists in order to evaluate the impact on children's play.

As application development for ubiquitous computing environments relies on software frameworks that provide higher-level abstractions, we have also investigated the design and practical development of software infrastructures. A prime example of these efforts is *Fosstrack* (www.fosstrack.org), an open-source software platform that facilitates business process automation using Radio Frequency Identification (RFID)

technology to automatically track and identify individual product items in the supply chain. The design and development of Fosstrack matured in the context of the Mobile and Ubiquitous Computing Lab (M-Lab), a joint initiative of ETH Zurich and the University of St. Gallen, Switzerland founded as early as 2001, whose goal was to bring the ideas of ubiquitous computing into companies that could benefit from the availability of smart devices, RFID, and related technologies.

Figure 2: Portable energy monitor

Smart Energy

The model behind the M-Lab, with its close cooperation with industry, also inspired the establishment of the *Bits to Energy Lab* (B2E) in 2007. Under the umbrella of the B2E, we are exploring the use of ubiquitous computing technologies to increase the transparency of the ecological footprint of economic and industrial processes, products, and services. In particular, we are investigating the applicability of sensor and actuator technologies for increasing energy efficiency, and also exploring new ways of influencing user behavior towards a more sustainable conduct.

An example of the projects promoted by the B2E is the eMeter, a portable energy monitor. The eMeter allows consumers to monitor their electricity utilization with their smart phones. At its core, the eMeter is an application for mobile devices that can retrieve the readings of digital electricity meters using a wireless connection. The information made available by the meter is then displayed to the user on the mobile device through a carefully designed interface. Users can thus learn about their total electricity consumption on a daily, hourly or even finer resolution. Also, information on the consumption of electrical appliances can be retrieved and visualized. This should make users aware of the presence in their households of inefficient, or thriftlessly used, devices.

Other recent activities also focus on the design and implementation of a smart heating control system for private households. The system will exploit information made available by existing smart devices, such as smart electricity meters and mobile phones, to estimate occupants' activity patterns and thus optimize the heating control strategy. By avoiding the need for installing customized sensing devices in the home, this should significantly lower the adoption barrier of smart heating solutions.

Contact

Further information can be found on the website of the Distributed Systems group at ETH Zurich: www.vs.inf.ethz.ch.

■

Announcements

EWSN 2012 – European Conf. on Wireless Sensor Networks

February 15 – 17, 2012, Trento, Italy

ⓘ http://ewsn12.disi.unitn.it/

Paper Submission Deadline: September 23, 2011

ARCS 2012 – Intl. Conf. on Architecture of Computing Systems (Focus: Platforms for Embedded Computer Systems)

February 28 – March 2, 2012, Munich, Germany

ⓘ http://www.lrz.de/~lis/arcs2012/

Paper Submission Deadline: September 30, 2011

MidSense 2011 – Workshop on Middleware Tools, Services, and Run-time Support for Networked Embedded Systems (at ACM Middleware 2011)

December 12, 2011, Lisbon, Portugal

ⓘ http://www.midsens.org

Paper Submission Deadline: August 15, 2011

RTSOAA 2011 – Workshop on Real-Time Service-Oriented Architecture and Applications

December 12-14, 2011, Irvine, USA

ⓘ http://link.eecs.uci.edu/conferences/rtsoaa2011/

Paper Submission Deadline: September 30, 2011

■