Sensors in the system cyle

Johan J. Lukkien
Department of Mathematics and Computer Science
Security and Embedded Networked Systems
Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

***Abstract*—It is generally recognized that creating an application for a wireless sensor network from scratch is time consuming and the task of specialists. What are the effects of technology advancements and which tooling is needed to improve this situation? Should we strive for 'IP to everything'? I think that there will remain room for application-specific communication technology but that there is a great deal of improvement needed in tools and languages to make the development process more manageable.**

I. Introduction

With the advent of programmable sensor and actuator nodes we also see a revival of the dawn of computer science. The challenges of dealing with low memory and processing resources remind me of the days when I worked with some 8 people on a PDP 11/34 mainframe running at a few MHz and having an addressable memory of 64K. However, the only network our PDP was connected to was the power grid. Further challenges of current sensor nodes comprise their radio communication, their numbers and the generally tight energy budgets.

An interesting aspect of the work on wireless sensor networks is that it reopens fields of research that were more or less closed because of established abstractions. In networking, MAC, network and application layer protocols are studied once again and application-specific alternatives are proposed. In Operating Systems, new mechanisms and programming models are proposed that better suit the need for programmer control. Writing C code in such a way that a compiler can generate concise and efficient machine code has become an art again.

It remains to be seen, however, that this approach is sustainable. The development cost associated with the current special-purpose nature of WSN application development is too large to be commercially attractive except for special-purpose deployments, for which the hardware development by itself is already costly. In addition, if WSN applications are to integrate with existing infra structure, standardization on protocols and development methods is mandatory. A good question is therefore to see what limitations and special approaches in WSN are intrinsic and which ones can be expected to disappear when time goes by, either because of better insight or because of advances in hardware.

II. The penetration of IP

A lesson from the last 15 years is the siege of the IP protocol. Not only did it emerge as the sole winner in the network protocol battle, it also survived an enormous upscaling and transformation of the Internet. The reason for this victory as I see it is that bringing a message towards a destination is a powerful interoperability concept. What the meaning of the message is can be decided by the receiver. We see indeed a strong divergence on top of IP. Every party not adhering to this IP convergence is 'out' in the worst possible way: it cannot communicate with the rest of the world. One must therefore have a very convincing story to propose an alternative and indeed, the concept of 'IP to everything', 'the Internet of things' and the like are being proposed today, with a focus in the design of 6LowPAN.

I think that establishing IP communication with small nodes in a point-to-point manner has important effects: it encourages developers to think about these nodes as communication end points and it changes the style of working, focusing on application layers on top of an (end-to-end) message passing stack. This is ok when nodes are powerful enough but it limits applications that integrate functionalities across layers in order to save resources like energy and time (latency).

As long as we have nodes that have limited memory resources for storing packets, limited energy resources for establishing end-to-end communication and a behavior of intermittent connectivity for a large percentage of the time we will need gateway technology at the network layer to make up for that. In other words, the standardization of the IP protocol would be nice but performance and other quality aspects cannot be hidden by it.

III. The effect of Moore's law

In the context of sensor networks we can interpret Moore's law along two lines. The first line is that while current price and size are maintained (or decreased slowly), more resources and higher speeds become available. This enables more costly abstractions than currently is done, like the full integration as IP endpoint and, perhaps, always-on communication. It also positions these nodes more as a part of the regular infra structure, the main difference being the absence of a local user interface and their, potentially, large numbers.

The second interpretation is that the cost of current nodes go down while maintaining roughly the currently available resource profile. A motivation for looking at such nodes in spite of technology adavances would be the energy budget which simply limits the amount of work that can be done. Among these are devices for which storing an IP packet or having the responsiveness expected of IP endpoints is not

feasible. This interpretation of Moore's law I regard as more challenging for the research community. When prices drop to the point of current RFID tags, massive deployment scenarios become indeed feasible.

IV. System integration

Moving IP technology into a sensor network is one option; the other option is to move the sensor network communication protocol into the IP domain. On the edge between sensor network and IP domain we then have a gateway that translates between the two formats. In this way we can have 'virtual sensor nodes' running as applications on workstations. The advantage is that this allows to leverage all energy optimization expressed in clustering and special MAC protocols into the sensor network while still having the integration with the IP domain. The virtual nodes are, in fact, application gateways.

A useful abstraction towards the Internet as I see it is given by the notion of services: the sensor network provides (and probably requires) a number of services that can be used by Internet nodes. Such services are integrated into larger applications using the regular SOA concepts of orchestration and choreography. The application gateway (that provides these services) should be designed such as to take away the peculiarities of the sensor network, translating these into quality properties of the service. For example, the fact that a temperature sensing node is available only one second per minute translates into a certain staleness of the temperature reading service. This is much better than having to poll for the existence of an IP connection.

V. Network programming

Sensor/actuator networks consist of larger numbers of nodes that together form the platform for an application. In which way should we program such a network? What are the development and deployment procedures and how does it fit in the software process? Again, there are a number of views and possibilities.

In 'regular' distributed systems, the deployment of an application involves the installation of a compiled code on each machine taking part in the final system deployment. This is usually easy since these machines have a) a powerful network connection b) enough foreground and background memory to store the application and c) an operating system that manages the application and deals with errors. This makes that the deployment part of a test-debug cycle is no real point of concern.

This situation is very different for embedded sensors that have a) a slow and unreliable network connection, b) must store their programs in ROM of which they have a limited amount and c) have limited OS functions and 'die' upon failure. An important tool to a developer is therefore an accurate simulator that emulates the node behavior exactly, including all external behaviors like sensing and communication. I think that an application should be tested completely integrated in a simulator before any deployment efforts are done.

Then, going to physical deployment the question is whether the used sensor network is to be regarded as a (frequently reusable) platform or as a one-shot system. The effort of handling a large number of nodes physically is only justified if it is done only once. Otherwise, installation procedures must be present to install the application over the air. This is especially challenging since this installation includes the definition of communication and may even include the definition of the used packet format and timing.

A most important question though is whether we should think about an application as the resultant of local behaviors that we put on each node or that we want to regard it as a single application for the network having local behaviors as a derivative. I think we should design applications without thinking about particular nodes but while thinking about services we want to have delivered. The language should give us the right concepts for this and the compile-deploy cycle should hide the details and establish the IP service integration at the same time. This is, of course, an abstraction, and abstractions cost performance. It is a challenge to limit this performance loss by finding the right programming concepts. These should be rich enough to describe local and distributed computations as well as effective messaging, supporting the particular application at hand. Managing the distributed resources and taking resource-based decisions must be expressible as well. In particular, the programming concepts must make it possible to express computations and communications that remain local to the sensor network and are not all delegated to the IP domain.

VI. Summary

As time goes by also small embedded devices like wireless sensors and actuators will have more resources. This makes it possible to integrate them directly with the regular infra structures. There will remain room, however, for devices and applications with specialized hard- and software. Service oriented architectures are useful to integrate these with IP infra structure. Programming concepts must be investigated and developed to program the network as a whole in a cost-effective manner. Resource-management should be an integral part of this. Such concepts are useful to program large numbers of the first class of devices as well.