UNIVERSITY *of* VAASA

REINO VIRRANKOSKI
(Ed.)

Generic Sensor Network Architecture for Wireless Automation (GENSEN)

PROCEEDINGS OF THE UNIVERSITY OF VAASA

REPORTS 174

VAASA 2012

Aalto University
School of Electrical
Engineering

UNIVERSITY of VAASA

Seinäjoen ammattikorkeakoulu
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

TEKES

PREFACE

This is the final report of the project *Generic Sensor Network Architecture for Wireless Automation* (GENSEN). The project took place on 2009–2011. It was funded by participating companies and research organizations and by the VAMOS-program of the Finnish Funding Agency for Technology and Innovation (Tekes). Participating research organizations were Department of Automation and Systems Technology and Department of Communications and Networking in Aalto University, Communications and Systems Engineering Group of the Department of Computer Science in the University of Vaasa, and School of Technology and School of Agriculture and Forestry in Seinäjoki University of Applied Sciences.

AUTHORS

Emre Ilke Cosar	Aalto University
Maurizio Bocca	Aalto University
Mikael Björkbom	Aalto University
Shekar Nethi	Aalto University
Aamir Mahmood	Aalto University
Lasse Eriksson	Aalto University
Riku Jäntti	Aalto University
Heikki Koivo	Aalto University
Huseyin Yigitler (Yusein Ali)	University of Vaasa, Aalto University
Simo Keskinen	University of Vaasa
Niklas Wik	University of Vaasa
Faheem Ahmed	University of Vaasa
Matti Tuomaala	University of Vaasa
Reino Virrankoski	University of Vaasa
Petri Hänninen	University of Vaasa
Mohammed Elmusrati	University of Vaasa
Heikki Palomäki	Seinäjoki University of Applied Sciences
Marko Huhta	Seinäjoki University of Applied Sciences
Matti Tassi	Seinäjoki University of Applied Sciences
Petteri Mäkelä	Seinäjoki University of Applied Sciences
Antti Pasila	Seinäjoki University of Applied Sciences
Jussi Esala	Seinäjoki University of Applied Sciences

Contents

1 INTRODUCTION

There has been a remarkable development in the area of wireless sensor networks (WSNs) during the latest decade. This development has also opened new opportunities for wireless automation. Wireless network enables access to such places that cannot be cabled, such as rotating parts of the machine and mobile machines. Wireless access makes it also possible to remarkably increase the number of measurement points to get more rich data from the system for monitoring and control purposes. Sensor nodes equipped with one or several processors enable advanced distributed data processing, local control loops and other intelligent operations in the network without centralized computation.

In this project we focused strictly on IEEE 802.15.4 communication protocol and 2.4 GHz industrial, science and medical (ISM) band. This communication standard is the most common in the context of WSNs, and the unlicensed ISM band is open for all users. We created a generic software and hardware architecture for wireless automation, and verified its feasibility by building five different pilot applications from different application areas: industrial environment, wind turbines, distributed energy production, greenhouse and cattle house. The business potential of the developed system as well as steps to commercialization was also considered in the research work. The project consortium consisted of two universities, one university of applied sciences and seven companies. It was funded by Finnish Funding Agency of Technology and Innovation's VAMOS-program (Tekes).

2 CURRENT STATE OF THE ART

2.1 Background

There have been some cable replacement techniques, such as remote controllers, in use already for decades, but the rapid development of wireless networks has happened during and after the development of cellphone networks. Many of the earliest projects at the beginning of 2000's were funded by military. Some of the comprehensive ideas there was to develop low-cost, miniaturized sensor nodes, which can be scattered to extract information from large areas to be able to produce a situation model which has a supreme accuracy compared to the enemy during battle or intelligence operation. The scarce communication, memory, computation and power resources of the sensor nodes have presented challenges for the research since then. Moreover, the distributed nature of the network required the development of such networking protocols, which can be executed in the nodes in a distributed manner without continuous involvement by any centralized controller.

When the research proceeded during the latest decade, many applications outside the military sector also started to emerge. One of the application areas having a remarkable business potential is wireless automation. As indicated by Figure 3.3.1, the value of the industrial WSN markets on 2011 is estimated to be nine times the value it was on 2007.

2.2 Requirements Rising from the Application Area

Wireless sensor network consist of sensor platforms called senor nodes. These platforms have at least microprocessor or microcontroller, some memory, radio, power supply and one or several sensors. The nodes can communicate directly with each other and they can route messages between such nodes, which are not directly connected to each other by a single radio link, but which are connected by a path formed by several radio links. There can also be actuators equipped with the same type of radio so that they can operate in the same communication network with the sensor nodes. Because of this, a name wireless sensor and actuator networks is also used quite often.

In the context of wireless automation, WSN usually operates as a part of the automation system. As a consequence, its interfacing and compatibility with other parts of the system must be considered. The general requirements of the automation system must be filled also in WSN. Some of the most critical ones are system performance in terms of data transmission rate and sample rate, communication reliability and power supply. The last one is especially critical in the context of WSNs, because the network must be able to operate long periods without a need of such system service, which would require human involvement. In addition to automatic network initialization, self-diagnostic and self-healing operations, it is also important to be able to perform intelligent operations in the network in a distributed manner. There is no challenge or technical novelty in the pure cable replacement anymore, but once the measurement data can be processed in the network such that only relevant information will be transmitted to the upper levels, and once the networks can adjust their operation in event based manner by taking advantage of the data they measure, completely new kind of automation solutions can be build. For example, more measurement points can be added to the system to be able to do more advanced control, and some of the measurement points can be added to such places which are not accessible in a wireless manner. Distributed computation enables also local decision making such that some control loops can be based on the local information without a need to cycle all information through the upper layers or through the centralized entities of the control system.

On the other hand, some of the WSN requirements presented in the context of military applications are not that dominating in wireless automation. Instead of having a size of so-called Smart Dust in the magnitude of cubic millimeters or their fractions, the size of sensor nodes in wireless automation can be bigger. Low energy consumption is important also in wireless automation, but filling the system performance requirements is more important. This may effect on the selection of applied microprocessors, memory, power supply and other electronic components. The protocol software must also be designed in such a way that the performance requirements are filled. However, different applications may have dramatically different performance requirements. For example, in structural health monitoring or in process automation we may be on the limits of the sample rate and data transmission capability, but in greenhouse automation it may be well enough to take one sample in 10–15 seconds. The possibility to adjust the performance according to each particular system requirements must be included to the protocol software to make the system generic.

2.3 Standards

In this project we focused strictly on IEEE 802.15.4 communication protocol operating in 2.4 GHz ISM band. Most of the research and development activity in the area of WSNs has been focusing on IEEE 802.15.4, and 2.4 GHz band is globally unlicensed. However, the joint use of IEEE 802.15.4 and IEEE 802.11 (WLAN) is also considered as well as network interfacing to the other parts of the automation system.

The most important existing standards in the area of wireless automation are WirelessHART [1] and ISA100.11a [2]. In April 2010, Wireless HART was approved by International Electrotechnical Comission to become the first wireless standard as IEC 62591. Even though the WirelessHART standard itself is open source, it is based on Time Synchronized Mesh Protocol (TSMP), which is licensed to Dust Networks Inc. [3]. The requirement to achieve faster duty cycle simultaneously with more light weight operation in terms of energy, computation and communication as well as options for license free software development and communication make it reasonable to develop other IEEE 802.15.4 based solutions in addition to WirelessHART and ISA100.11a. However, these two standards need to be considered.

2.4 Towards Generic Solution

So far the existing wireless automation solutions made by using wireless sensor networks operating in 2.4 GHz band under IEEE 802.15.4 communication protocol have been case specific. Even though there have been some ideas about generic solutions, the reported cases have typically included a lots of hardware and software development required separately for each application. Most of the required sensors exist already, but the challenging part is to integrate them easily to the wireless systems. There has been also a lot of research in the protocol development related to network operation, such as self-configuration, self-healing, localization, routing, energy efficient operation, data compression, data fusion etc. However, there is still work needed to be able to build such protocol software, which includes the most important features for different applications and enables different operational modes depending on the application needs, but is also feasible to be used with the scarce resources of wireless sensor nodes.

The need for such a generic software and hardware solution, which includes all important functionalities in the protocol side, enables distributed computation and data processing in the network, enables interfacing with other parts of the automation system and allows us to easily implement different commercial sensors to the system, was identified as a key target in GENSEN project. In terms of time and performance the targeted solution is illustrated in Figure 2.4.1. Developed solution provides a platform to rapidly produce several kinds of customer specific cases [4, 5]. System business potential is discussed in Chapter 3, and design process in Chapter 4. Developed software and hardware architecture is described in Chapter 5. System testing and validation with five different applications is presented in Chapter 6 and finally conclusions in Chapter 7.

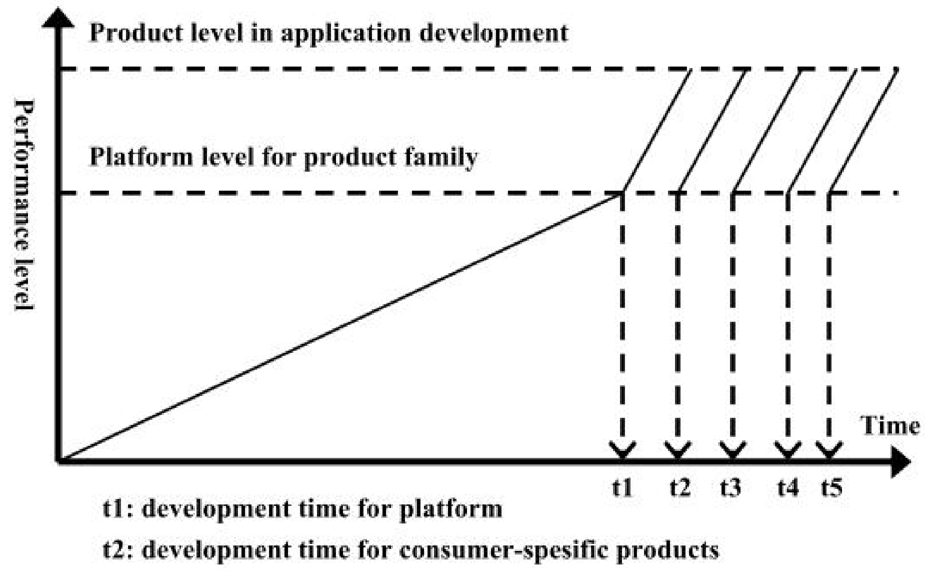

Product level in application development
Platform level for product family
Performance level
Time
t1
t2
t3
t4
t5
t1: development time for platform
t2: development time for consumer-spesific products

Figure 2.4.1. Developed generic platform enables fast building of different applications [4].

3 BUSINESS POTENTIAL

3.1 Business Concept and Models

The design process of the business model is presented in Figure 3.1.1.

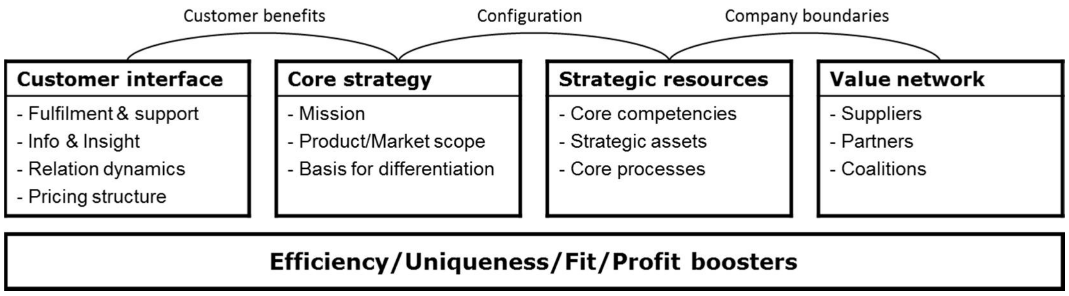


Figure 3.1.1. Business model design process.

The business model as well as business concepts can be observed from different points of view by placing, for example the following questions:

- What is the internal logic and way to proceed in the design process?
- How the actual product and the service potential it provides is taken into account in the business concept design?
- What are the sub-entities of the business model, what are their roles in the design process hierarchy and what is the typical context?

A business model based on the multi-purpose platform is illustrated in Figure 3.1.2. The structure and utilization of the business model is presented by Figure 3.1.3.

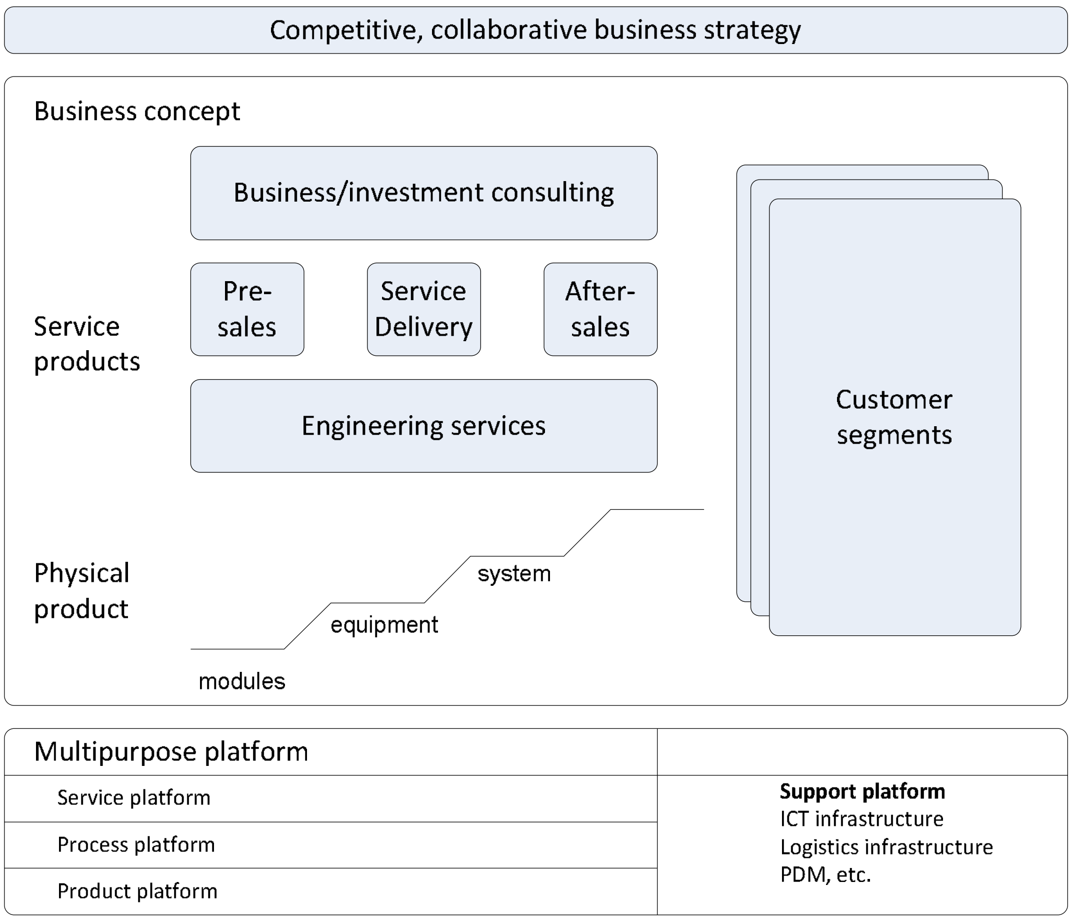


Figure 3.1.2. A multi-purpose platform based business concept.

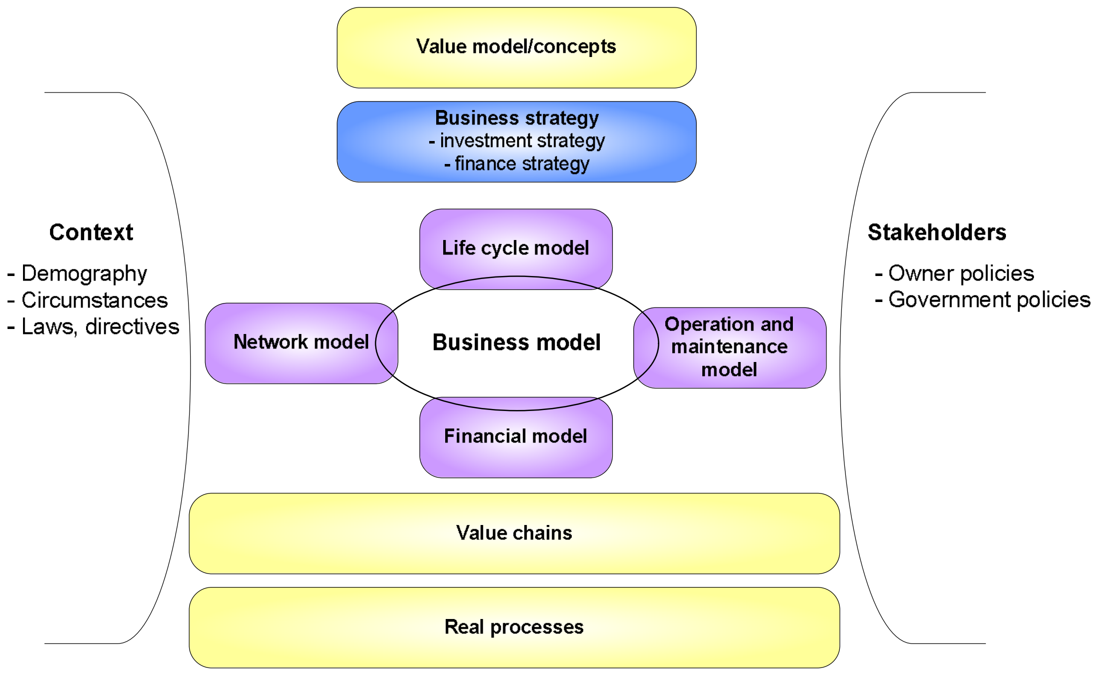


Figure 3.1.3. The structure and utilization of the business model.

A system level approach is required to be able to efficiently design technology and business in the context of wireless sensor networks (WSNs). The approach is discussed in more detail in Chapter 4. In the context of GENSEN-project, the evaluation of the developed system business potential was one of the main tasks. The business potential was evaluated by making a literature overview and by interviewing the key industrial and research personnel in Finland. The way how the business potential consists of different sub entities is illustrated in Figure 3.1.4.

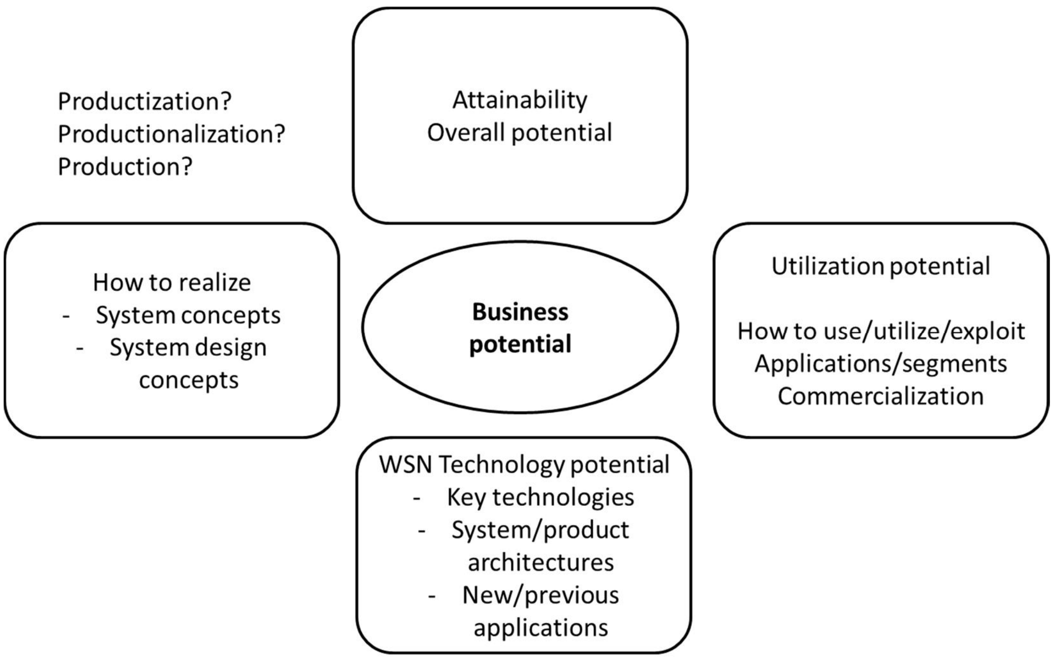


Figure 3.1.4. The business potential of WSNs.

Figure 3.1.4 was shown to each interviewed person as a part of the preparation material to direct the discussion to the main issues. As indicated by the uppermost block in Figure 3.1.4, the results of the analysis and synthesis will provide the answers to the strategic questions regarding how much can be achieved and managed and what is the total business potential. Left and right blocks indicate how the total potential can be utilized. Left block stands for the product selection and system design of the producing and marketing company. The process is divided to several phases starting from application description to system planning and design and finally ending up to commercialization. Right block stands for the recognition of the utilization potential. In big shipments the services and support is today a remarkable part of the business. Its share of the annual exchange can be around 30-40%.

3.2 Planning Product Ranges, Generations and Development Processes

Product range planning is based on company strategies regarding business, target customers, products and technology. The fundamental starting questions at the beginning of the product range planning process are:

- What kind of products will be offered to different customer segments, customers and application areas?

- How market-based thinking (market pull) and technology-based thinking (technology push) can be combined in most productive way?

Product planning can be presented by using different production portfolios, product-market matrices and product roadmaps. General levels of the product planning are presented in Figure 3.2.1.

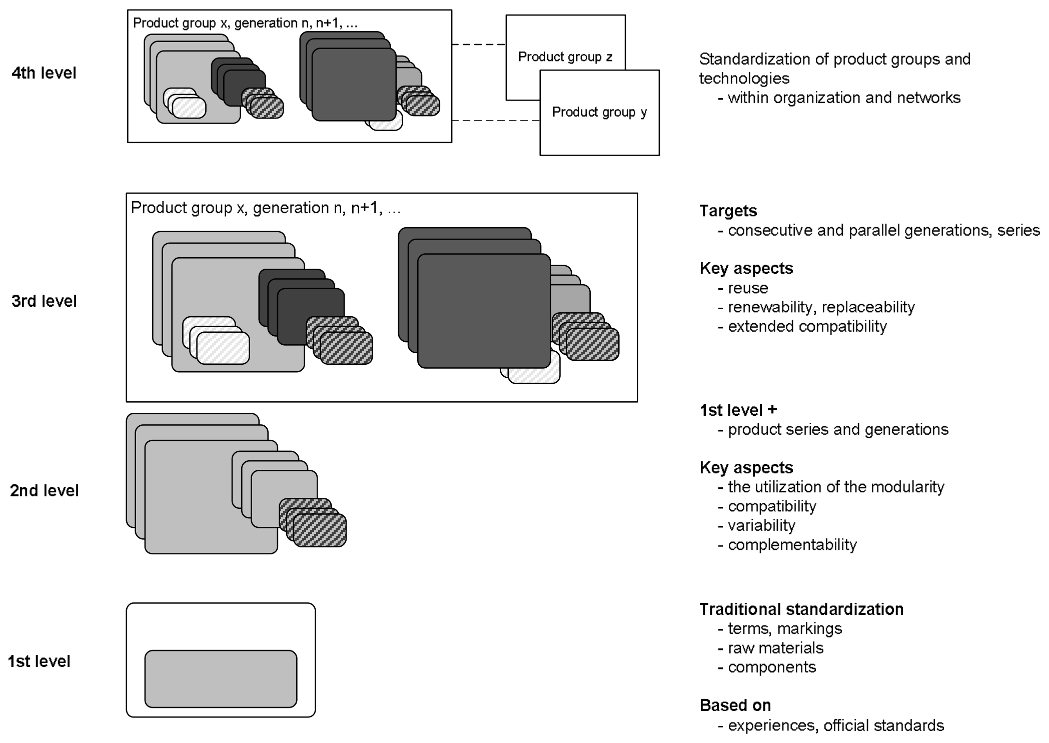

4th level
Product group x, generation n, n+1, ...
Product group z
Product group y
Standardization of product groups and technologies
- within organization and networks
3rd level
Product group x, generation n, n+1, ...
Targets
- consecutive and parallel generations, series
Key aspects
- reuse
- renewability, replaceability
- extended compatibility
2nd level
1st level +
- product series and generations
Key aspects
- the utilization of the modularity
- compatibility
- variability
- complementability
1st level
Traditional standardization
- terms, markings
- raw materials
- components
Based on
- experiences, official standards

Figure 3.2.1. Product planning process.

In Figure 3.2.1, the lowest level represents the traditional single product, or size-scaled single product set, planning process. Next level introduces the principle of modular thinking, which rises from the system level approach. In modular thinking, the key idea is to develop and produce such building blocks, which can be freely combined to build different systems based on the particular needs. As a consequence, one can target to minimize the number of different modules which are required to build different products and in the best case the entire range of products offered by the company. This is a very efficient method in many industrial areas, and it is well applicable to WSNs.

Software technology has offered new opportunities to apply the principle of modularity, but on the same time it may also set some limitations. Physically different hardware components can be made compatible by software interfacing, but on the same time one must also worry about the compatibility in terms of software architecture. The latter includes also backward and forward compatibility between different product generations. Once the modules are renovated, a careful planning is required in advance to ensure that all compatibility requirements are filled. This addresses to platform thinking and architecture.

The immaturity of the existing license-free open source wireless automation solutions, which are based on IEEE 802.15.4 communication protocol, has been indicated by the fact that even though there are a lot of sensors for different industrial measurements commercially available, the software and hardware solutions have usually been separately tailored for each application. In GENSEN project, a generic hardware platform and a generic protocol software was developed to fill this gap. They are explained in more detail in Chapter 5.

The fourth level in Figure 3.2.1 can be, for example, the compatibility between products and systems produced by a big company or company alliance. The entire process from feasibility study to production start-up is presented in Figure 3.2.2, and an example of the platform architecture in Figure 3.2.3.

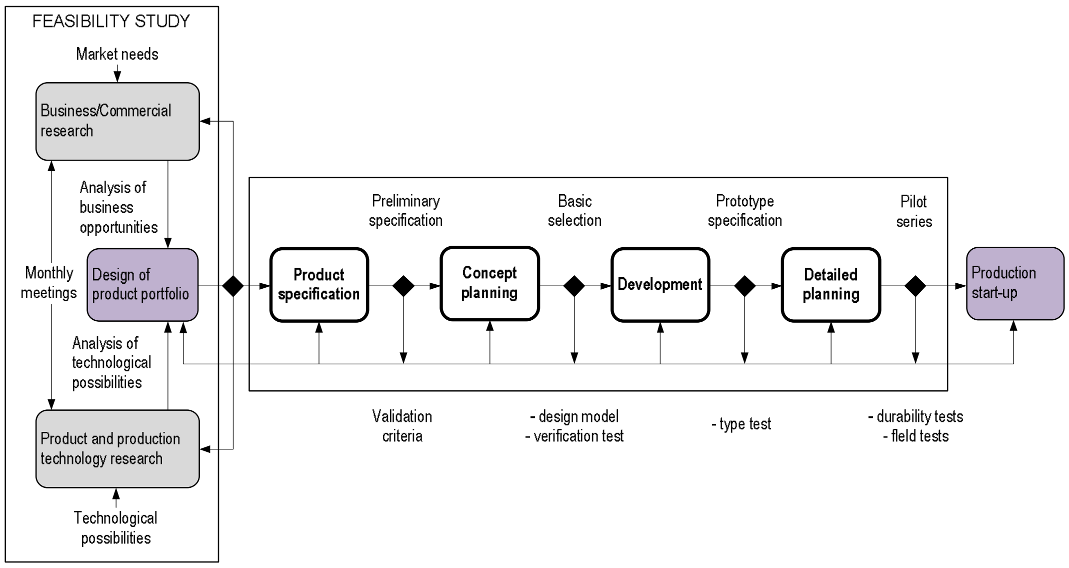


Figure 3.2.2. A process from feasibility study to production [6].

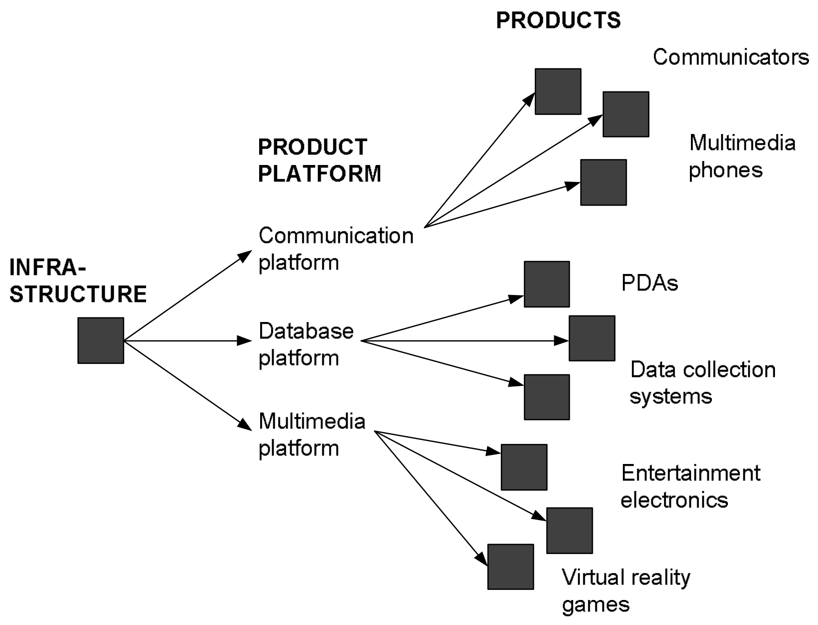


Figure 3.2.3. An example of the product family based on platform architecture.

Products and their development projects are interconnected in many ways. These interconnections are presented by the product-process matrix of Wheelwright and Clark [7] in Figure 3.2.4. It can be seen from the figure that new so-called breakthrough projects are forming their own subgroup. The rest of the product development projects can be divided to two main groups; platform projects and derivate projects. Close to derivate projects are also so-called customer variant projects, which deal with modifications required by some particular customers.

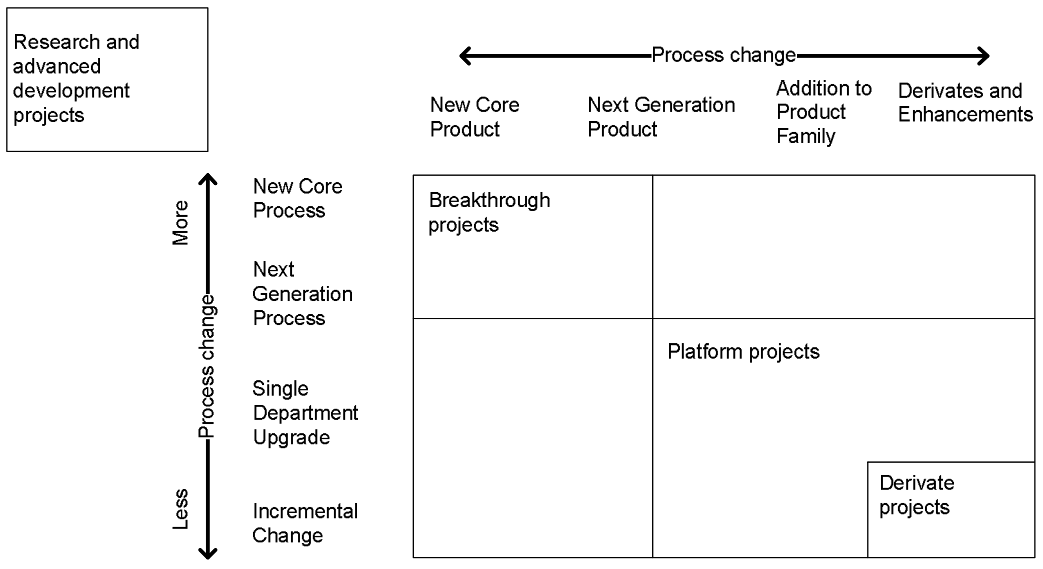


Figure 3.2.4. Dependencies between product development changes [7].

Regarding the product range and the development processes based on the platform thinking it is stated: *“For complex systems such as mainframes or military systems in particular, designing and evolution process into the system concept itself may allow easier subsystem improvements over time and thus maintain a high degree of communality from one product generation to the next while decreasing the life cycle of each generation and adjusting more rapidly to the evolution of markets”*.

There are several major trends producing larger point of view for continuing, dynamic conceptualizing, constructing, utilizing and updating platforms in the companies working in ICT area. Questions, which are continuously reasonable, are:

- From which elements the platforms consist on and what are their architectures?
- Where and how the platforms can be used?
- What are the impacts and advantages which can be reached?
- Where are we right now, what requires further research?

3.3 Commercialization

Following factors must be taken into account in product commercialization:

- Product lifecycle
- A moment to announce the product and enter to the markets
- Market studies
- Customer satisfaction
- Customer investment decision point of view

A timing to announce a new product, or product version, and enter to the markets is remarkable factor in the business competiveness. The pressures rising from the timing have also shortened the time ranges in the product development. In the case of wireless sensor nodes, the product lifecycle has been relatively short so far. As a consequence, it would be important to take care of the compatibility between old and new versions (especially the backward compatibility of the new ones) of the sensor nodes. Also a well-defined interface between protocol software and such a part of the software which is connected to certain hardware is important, because it enables two things. First, the well-defined interface makes it possible to change flexibly from one protocol software to another without touching to that part of software which deals with a certain hardware. This makes it possible to use different versions of the same protocol software or to use different protocol softwares (or protocol stacks) with the same sensor platform. Second, if the hardware is changed from older to newer version, the same protocol software can still be used. The increase of the world markets of industrial WSN applications is presented in Figure 3.3.1.

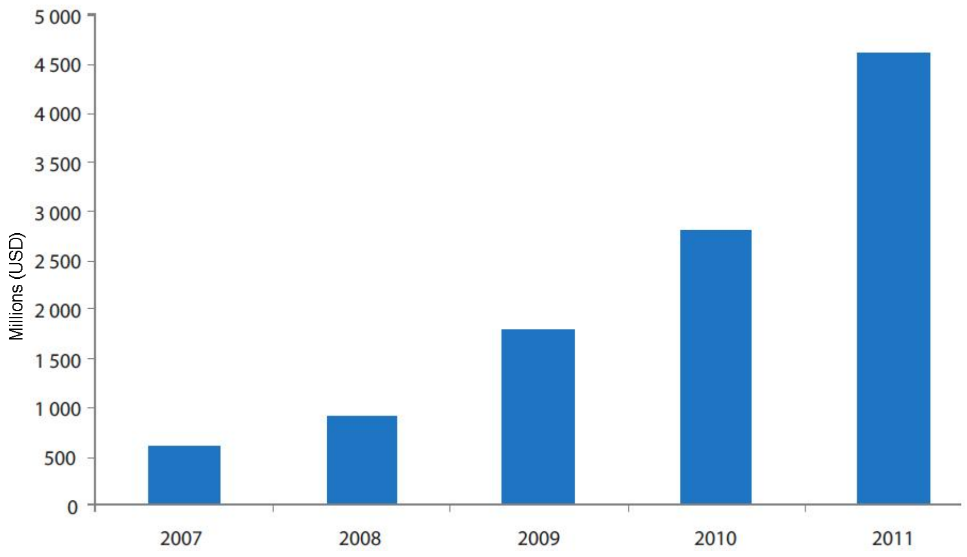


Figure 3.3.1. The increase of the world markets of industrial WSN applications.

During the first decade of WSN research, a remarkable part of the research was focusing on low level issues, such as data transmission protocols, energy efficiency and sensor node localization. These issues are mandatory to solve before entering to the markets, but solving them is not enough. The relation and interfacing between WSNs and other parts of industrial automation system as well as customer interests (willingness to invest) and customer satisfaction must also be considered. The development of standards must also be followed such that the customer requirements related to standards and possible commercial technology convergence towards certain standards will be realized well in time. In the case of wireless automation, the most important standards are currently WirelessHART and ISA100. However, in addition to them there is room for such a solution which enables multi-hop networking with intelligent measurement, control and data processing operations in the network. Bottlenecks in system feasibility and performance are caused by sensor node power supply solutions and by WSN's communication, memory and computation resources which are scarce compared to cabled systems.

The decision making model in customer's investment decision point of view is presented in Figure 3.3.2. In the case of WSNs, the case-specific criteria which is needed in the current state of the art of WSN's, is a systematic testing period which must be long and complete enough to ensure the system feasibility and reliability.

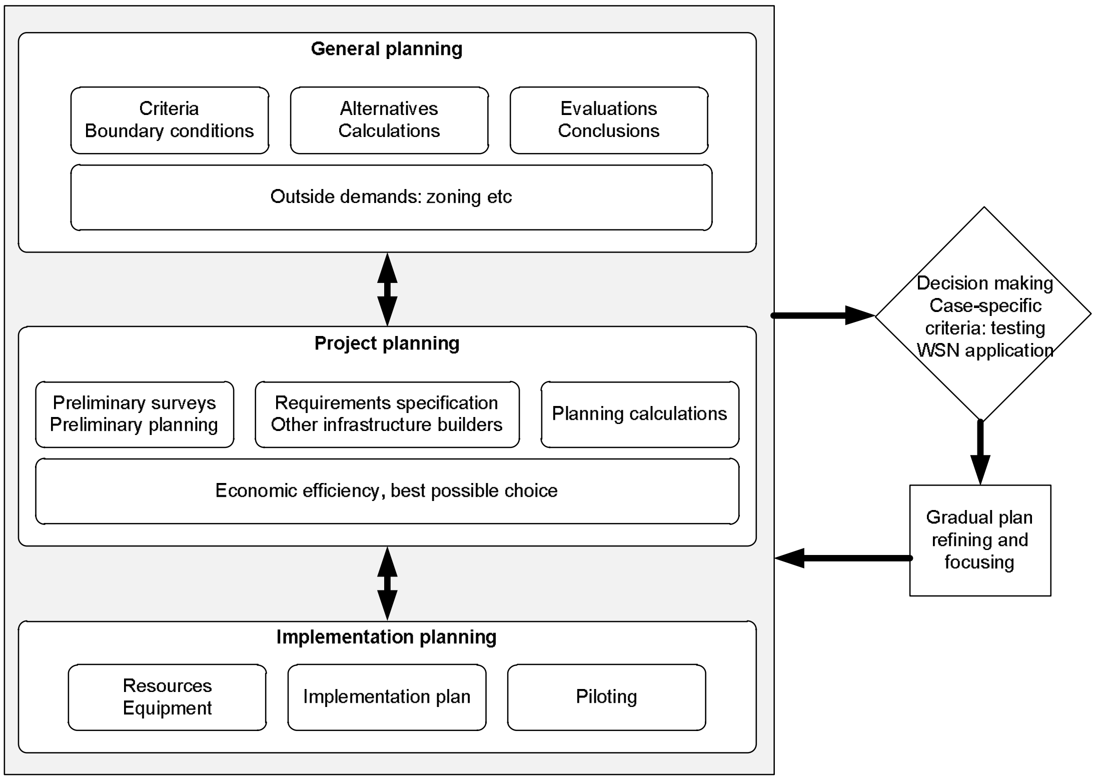


Figure 3.3.2. WSN investment decision making in customer's point of view.

4 SYSTEM LEVEL APPROACH

4.1 General

In real automation systems the wireless automation rarely operates alone. Usually it is interfaced to the other parts of the automation system. The wireless communication architecture itself can include several levels and several wireless communication protocols. Network can consist of several kinds of sensor nodes and actuators, which can execute different tasks based on their resources and equipment. Measurement data collected by wireless sensor nodes can be aggregated and processed in many abstraction levels. In control engineering point of view, there can be simple control loops, MIMO-systems and hierarchical control structures. It is not enough to optimize the operation of individual devices to make the system run properly, also the hierarchical system level design must be applied.

The practical basic questions concerning both existing and new system to be created are:

- What is the entity, what are its parts, elements and subsystems?
- What are the system boundaries?
- What is the operational environment?
- What are the structures in a system and between systems, e.g. physical and functional ones?
- How the system operates?
- Who are the interest groups or stakeholders?
- How to describe, visualize, model, design, simulate, realize, use and update the system?

4.2 System Concept

As a highest level of abstraction, a systemic environment of technology companies is presented in Figure 4.2.1.

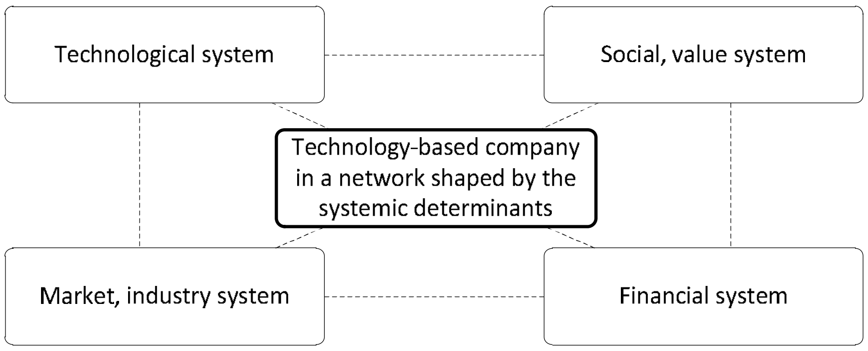


Figure 4.2.1. The systemic environment of technology-based company.

A framework for technology management carried out by the case studies in Finnish electronics companies is presented in Figure 4.2.2. As indicated by the framework, the best outcomes in technology management are competitive product family, strong collaboration, efficient general strategy and good work culture in the company.

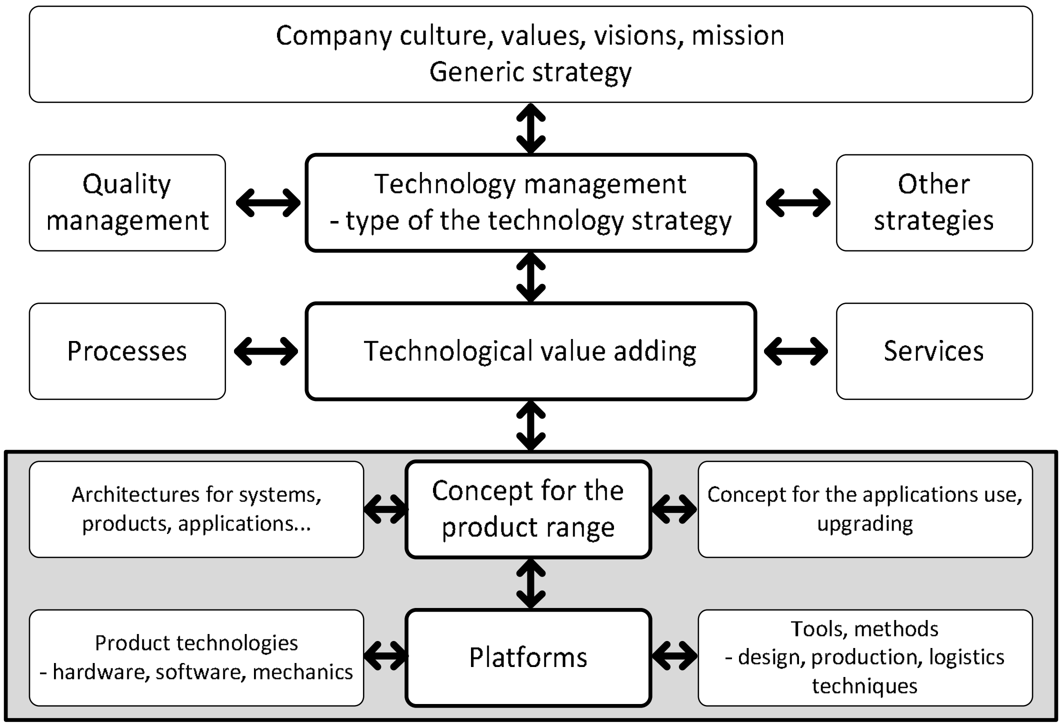

Company culture, values, visions, mission
Generic strategy
Quality management
Technology management - type of the technology strategy
Other strategies
Processes
Technological value adding
Services
Architectures for systems, products, applications...
Concept for the product range
Concept for the applications use, upgrading
Product technologies - hardware, software, mechanics
Platforms
Tools, methods - design, production, logistics techniques

Figure 4.2.2. A framework for technology management.

The way how the general principles of system concept are applied in GENSEN, is presented in Figure 4.2.3. The counterparts in the vertical axis are crucial. Analyzing, diagnosing and synthesizing the issues mentioned there is strategic level processing. To be able to successfully process it, one must be capable to see the big picture and recognize the important things from it. In the case of WSNs it means the ability to detect the generic features which make the sensor network architecture capable to fit to as many types of applications as possible. Moreover, part of the generic architecture is also the possibility to interface the sensor network easily with the other parts of the automation system. The uppermost part of the top axis counterpart indicates that the better the invited solution is the better is also the possibility to identify the entire synergy potential. Synergy cannot be created separately. It rises during the actual work as well as good quality. However, the identification of the sources of synergy is required.

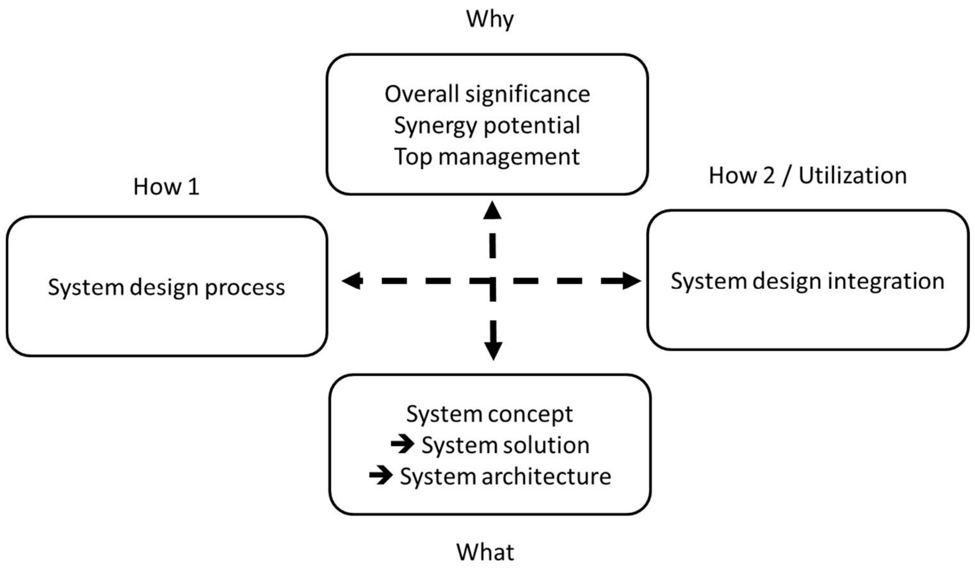


Figure 4.2.3. Applying system concept in WSN desing.

The counterpart in the horizontal axis stands for the fact that WSN integration to the other parts of the automation system (interfacing, compatibility etc.) must be taken into account in the early levels of the system design.

4.3 System Design Process

The co-design of software and hardware components is a mandatory approach in the case of embedded systems such as WSNs. Every organization can have their unique features in the design process, but in general the processes follow the pattern presented in Figure 4.3.1.

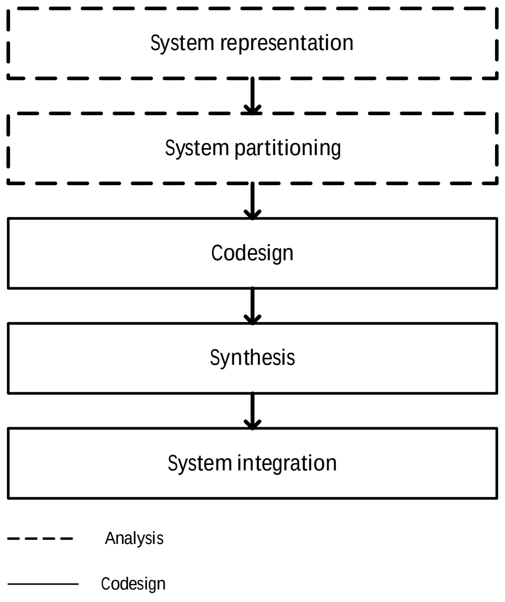


Figure 4.3.1. The general pattern of embedded system design process.

More specific model describing the design process of WSNs is presented in Figure 4.3.2. It can be applied to both case specific design projects and also to platform project as a part of industrial product development process.

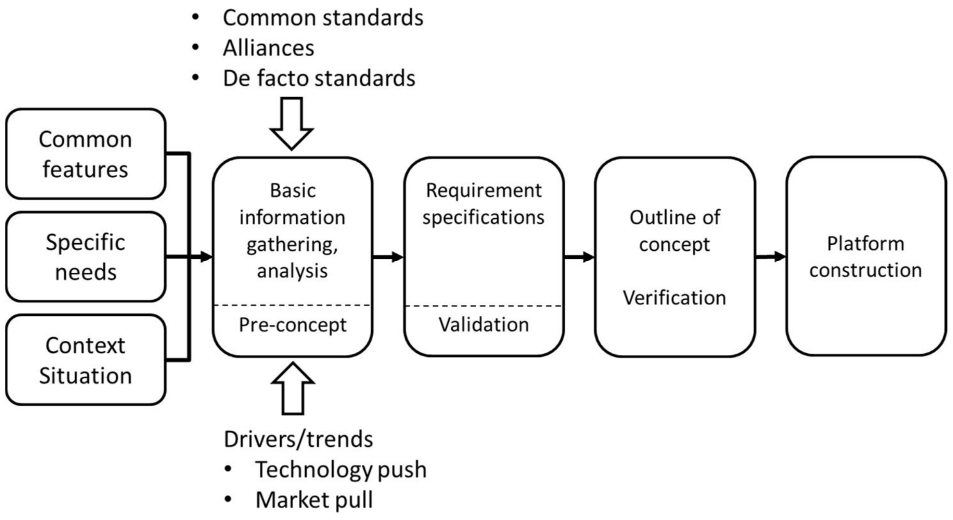


Figure 4.3.2. The design process of WSN.

Common target features of industrial WSNs completed with the particular application needs in the targeted customer area provide a starting point for the WSN development process. In industrial automation, typical critical requirements are communication reliability, system ability to fill the real-time requirements, system performance in terms of sampling rate and data transmission capability, sensor node power supply and the length of the reliable communication range a single sensor node can achieve when filling the defined requirements. In addition to pre-mentioned requirements, a set of features which are increasingly interesting, especially in the case of WSNs operating under IEEE 802.15.4 and IEEE 802.15.4a communication protocols, is the amount of intelligence that can be added to the network. Since the amount of measurement data is continuously increasing, it would be reasonable to pre-process and compress the data in the network such that only that part of information, which is needed in monitoring and control, is transmitted to the upper levels. Local intelligence in the wireless actuators and sensor nodes makes it possible to create local control loops and local network responses. These functionalities enable a higher degree of distribution in automation systems as before. However, new features can be applied only if the basic performance and reliability requirements are still filled.

4.4 Interviews

The interviews utilized in this market and commercialization analysis were made during Spring and Summer 2011. Altogether 18 experts from 12 different companies, universities and universities of applied sciences were interviewed. In each interview the current state of the art, the factors with remarkable impact and views about the future were all discussed. It was quite a common view that the general development phase is currently in the area of rapid growth in the hype curve. That area includes also some extra enthusiasm with some unrealistic expectations. According to another well-known presentation by Geoffrey Moore [8], the area of wireless automation is currently experiencing a rapid growth which also includes some turbulence. In such a phase some old companies are typically dying and some new ones are born. The ones which can keep their business running in the competition will produce such a growth of the sales which will lead to the breakthrough of the new technology. One common view indicated by several interviews was that in terms of technological maturity the RFID technology is several years ahead of WSN technology. Their hybrid solutions are expected to exist in the nearby future.

It is also expected that the standardization will play the most remarkable role in the development of the whole WSN technology sector. ISA and IEEE standards are considered equally strong in challenging real time industrial applications. The selection between them, and possibly some other options, will be defined by each particular application environment with its customer requirements. Some traditional solutions are also expected to be replaced by wireless field buses. Most probably different coalitions and licensing strategies will solve the competition between de facto standards.

Additional factors, which are expected to play an important role in the development, are the availability of the key components, algorithm research and development work, and in some cases also the easiness of the network components production and installation process. Different business opportunities are not clear yet, but the increase of subcontracting based product development and services is expected. More companies focusing on system integration will probably exist in the nearby future.

Crucial requirements for WSN specification model in the area of wireless automation pointed out in the interviews are

- Application area and operation environment
- Network mobility type: Static (non-mobile) or mobile network
- System performance requirements: sample rate, data transmission capability, real-time performance capability, data transmission reliability
- Sensor node power supply and power consumption
- What are the other functions and characteristics that need to be considered?
- Management of product portfolio: reusability, scalability and configurability
- Life cycle management, maintenance and expandability
- Overall system safety and reliability
- How to interface the WSN with other parts of the automation system?
- Installation
- Is there existing design tools and need to develop new ones?
- What are the requirements rising from production and distribution logistics?
- How to make the product complete for marketing?

Certain application areas of wireless sensor networks can be a potential niche-area for some companies in Finland. So far the recognition and utilization of such relatively narrow market slots have created some success stories. However, the wider views of the future of wireless automation are still shadowed by technological shortsightedness.

5 DEVELOPED ARCHITECTURE

5.1 Protocols

5.1.1 A-Stack

A-Stack is a real-time protocol software for time-synchronized, multi-channel and slotted communication in multi-hop wireless networks. The software is developed to meet the reliability, latency and accuracy requirements of real-time applications such as wireless automation and wireless structural health monitoring and to provide a flexible development environment for such applications.

Building blocks of A-Stack are the radio transceiver and microcontroller unit (MCU), FreeRTOS real-time kernel [9], radio and timer interrupters, Medium Access Control (MAC) layer, Packet Manager, Service Manager, and Application Tasks running on the operating system. These building blocks are shown in Figure 5.1.1. Table 5.1.1 shows the tasks and their functionality in A-Stack. The real-time operating system used in A-Stack enables application tasks to be easily developed and re-used.

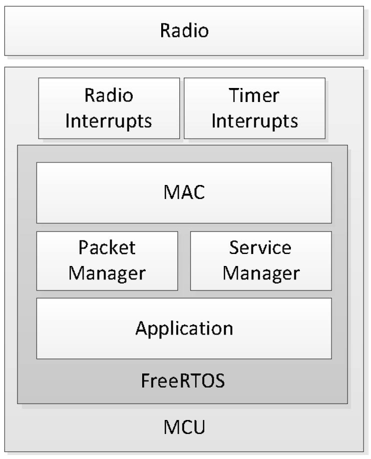


Figure 5.1.1. Building blocks of A-Stack.

Table 5.1.1. Tasks in A-Stack.

Task	Priority	Responsibility
MAC	Highest	Communication and timer events handling
Packet Manager	High	Incoming and outgoing packets handling
Service Manager	Normal	Initialization and reliability
Application	Low-High	Application dependent

A-Stack is independent of the schedule implementation. Thus, any schedule, including routing, communication pattern, and time-slot lengths, can be realized. IEEE 802.15.4 addressees are used, and unicast packets are acknowledged within one time-slot. The stack includes MAC, routing and time-synchronization protocols. It also provides online configuration and network reliability tools such as node joining and re-joining services as well as dynamic channel hopping. The stack is further supplemented with PC tools for optimizing the network as per the target application for easy prototyping.

In A-Stack, a communication pair (CP) structure is used to store the information needed for data exchange between the nodes, i.e. the address and network id of each node pair. Communication pairs and timer events are created separately and communication pairs are assigned to the timer events. In order to prevent congestion related problems, every communication pair is assigned a separate data queue. By separating the queues, it is possible to track the number of retransmissions for every link and prevent spreading congestion that occurs in one link. This allows multiple flows to take place simultaneously.

An example network topology and schedule generated for this topology is shown in Figure 5.1.2. Numbers under the *TX* and *RX* slots indicate the radio channel used for that particular time-slot. Note that multi-channel operation takes place in several time-slots. Time slots marked with S_T, S_R and S_I are used for service messages, node joining, network configuration and synchronization. Detailed information on the stack and its performance measures can be found in [10] and [11].

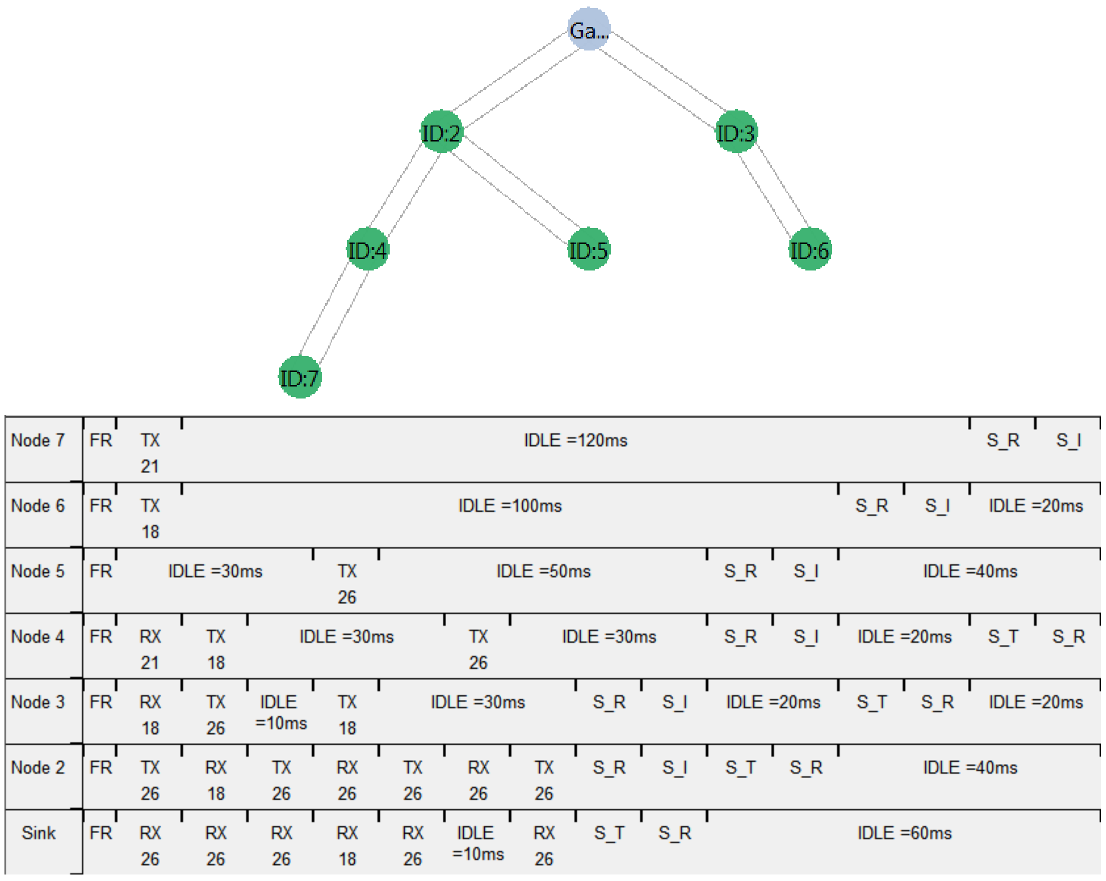

Ga...
ID:2
ID:3
ID:4
ID:5
ID:6
ID:7
Node 7 FR TX 21 IDLE =120ms S_R S_I
Node 6 FR TX 18 IDLE =100ms S_R S_I IDLE =20ms
Node 5 FR IDLE =30ms TX 26 IDLE =50ms S_R S_I IDLE =40ms
Node 4 FR RX 21 TX 18 IDLE =30ms TX 26 IDLE =30ms S_R S_I IDLE =20ms S_T S_R
Node 3 FR RX 18 TX 26 IDLE =10ms TX 18 IDLE =30ms S_R S_I IDLE =20ms S_T S_R IDLE =20ms
Node 2 FR TX 26 RX 18 TX 26 RX 26 TX 26 RX 26 TX 26 S_R S_I S_T S_R IDLE =40ms
Sink FR RX 26 RX 26 RX 26 RX 18 RX 26 IDLE =10ms RX 26 S_T S_R IDLE =60ms

Figure 5.1.2. 3-hop converge cast topology and schedule. Schedule length in terms of time is 155ms.

5.1.2 *Kick-Off Network Initialization*

Wireless sensor networks (WSNs) are composed of tens, potentially hundreds or thousands, of sensor nodes, i.e. low-power, resource-constrained, spatially distributed and battery-operated devices connected wirelessly, which can be equipped with sensors and actuators to cooperatively monitor and operate into the environment. Due to the high flexibility provided by the integration of sensing, computing and communicating capabilities into miniaturized hardware, WSNs are now being used in a large variety of applications, e.g. environmental and structural health monitoring, elderly and health care, security and surveillance, industrial process control, etc. In most of these applications, all the data collected by the sensor nodes deployed in the field must be timely and reliably transmitted to a central sink node, where they are eventually processed and stored. This communication pattern is called *convergecast*. For this purpose, the network is organized in a tree-like topology. After the initial deployment and before the start of the operation, i.e. data collection and transmission, a network initialization procedure aims at establishing a reliable communication infrastructure.

During the network initialization phase the deployed nodes gain knowledge of each other by exchanging neighbor discovery packets. All the information related to the existing connectivity graph can be collected at the sink node and processed in order to set up an optimal tree-like topology with respect to criteria like power consumption, latency and reliability.

The developed initialization procedure, named *Kick-Off* [22], achieves (1) neighbors discovery, (2) time synchronization of the nodes, (3) tree-like topology definition based on (4) links quality estimation and (5) symmetric key generation and distribution. Moreover, by adjusting its key parameters, the execution can be modified in order to minimize the network setup time, as required in military and emergency response scenarios, or to accurately assess the existence and quality of the links connecting the nodes, as required in environmental and industrial process monitoring applications.

The performance of Kick-Off is evaluated through deployments carried out in different environments, i.e. the inside of a university building, in an industrial hall, in a greenhouse, and in a cattle house. The case study results are presented in Chapter 6.

Kick-Off was first tested in university building, which was a typical indoor environment. Twenty sensor nodes were deployed in the area extending over a floor of the building. The testbed setup is shown in Figure 5.1.3. The performance of Kick-Off was evaluated over a total of 50 test runs, one in every 10 minutes. The nodes were kept in the same positions in all the time. To avoid the interference caused by the presence of four WLAN routers in the deployment area, the ZigBee channel 26 was used.

Figure 5.1.3 illustrates two tree-like topologies defined by Kick-Off. The presence of walls, objects, metallic surfaces, electric appliances, etc. in the deployment area makes the propagation of the radio signals unpredictable. For some nodes situated in particularly hostile locations, the parent selected by Kick-Off cannot be found in their close proximities. For this reason, the proposed network initialization procedure can also be used in order to assess, quickly and efficiently, if any of the nodes has been deployed in a communication black spot, i.e. in a region where the node is neither capable to successfully transmit to or receive from the other nodes of the network.

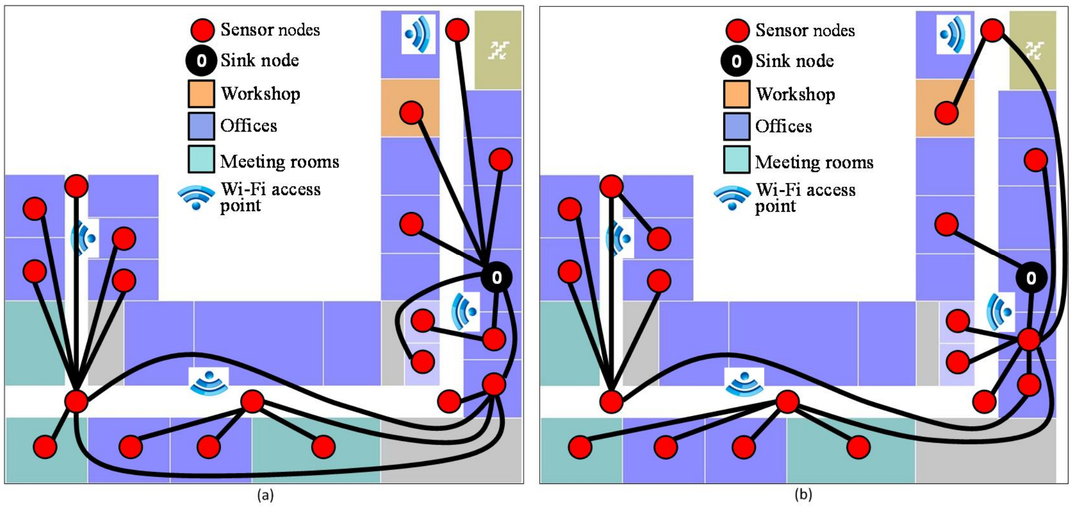


Figure 5.1.3. Two tree-like topologies defined by Kick-Off [22]. In (a), P_{TX} = 0 dBm; in (b), P_{TX} = -25 dBm.

In all the test runs, Kick-Off was able to successfully discover and poll all the deployed sensor nodes. However, regardless of the value set for the transmitting power of the nodes, Kick-Off was correctly able to select the high quality links over the worse ones. The average quality of the links selected to build the tree-like topology was higher than the one of all the discovered links. This is particularly critical with low transmitting powers, which could decrease the reliability of the network.

5.1.3 Reliable Networking Protocol

Designing an efficient communication protocol for industrial Wireless Sensor and Actuator Networks is a demanding task, since in control applications the data has to be delivered in a timely manner, while taking into account a dynamic network topology. We have developed a reliable MAC and networking protocol called Real-time Network Protocol (RNP). RNP is a collaborative TDMA/CSMA MAC and network-protocol designed especially for small, fast, and time-constrained WSNs operating in harsh industrial environments. Each node in RNP is dynamically allotted a transmission slot by the gateway. For retransmission either TDMA or CSMA operation is selected, which allows the network to adapt to changes in network topology and channel conditions and deliver the data quickly to the gateway. Moreover, RNP combines overhearing and piggybacking, which reduces communication loads and hence, improves efficiency. By using simulations we have shown that RNP outperforms alternative designs.

The protocol is designed for small networks (a few hops and up to 20 nodes) with tight real-time constraints of less than 1 s. The employed diversity techniques practically guarantee reliable operation, even in harsh industrial environments. RNP is based on limited broadcasting. Broadcasting is used in order to take advantage of mesh networking, which is robust to occasional link failures caused by fading and obstructions. Overhearing and piggybacking are used for forwarding missing data cooperatively. Uncontrolled broadcasting would lead to flooding and deteriorate the performance of the network. Therefore, a dynamic Time Division Multiple Access/Carrier Sense Multiple Access) retransmission scheme is used to control broadcasting and guarantee low latency.

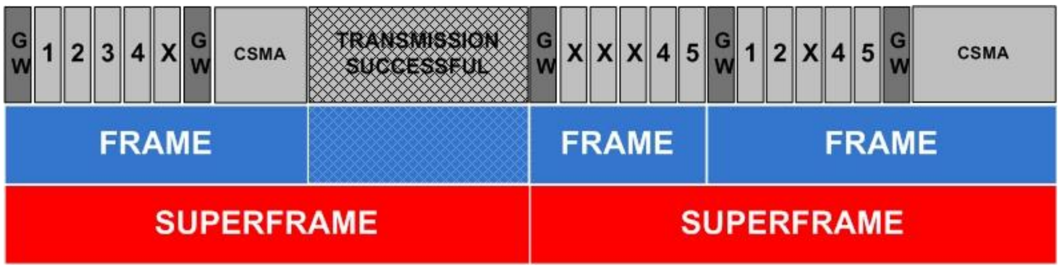


Figure 5.1.4. Schematic of the operation of RNP, numbers indicate the node ID. The gateway transmits a packet at the beginning of the frame. Transmission timeslots reserved for the nodes are highlighted.

All communication in RNP occurs during a superframe, see Figure 5.1.4, during which all the nodes should receive information from the GW and the GW should receive one packet from every node. A superframe consists of several shorter frames, which are divided into two phases: scheduled TDMA slots and random access CSMA. The length of the superframe is determined by the maximum delay tolerated in the system. The length of the TDMA phase is determined by the number of nodes, the rest of the frame is the CSMA phase, where any node can rebroadcast unsuccessful packets. All transmissions in RNP are broadcasts to allow for overhearing and piggybacking. Each frame is initialized with a broadcast packet by the GW. Once the GW has received one packet from each node, the network goes to sleep until the end of the superframe. Consequently, the length of a TDMA/CSMA phase is dynamically adjusted by the GW node depending on the network topology and channel conditions.

The RNP has been shown in both simulations and deployments in an industrial hall to perform with satisfactory reliability. The performance and reliability is considerably better than a simple polling strategy, as show in Figure 5.1.5.

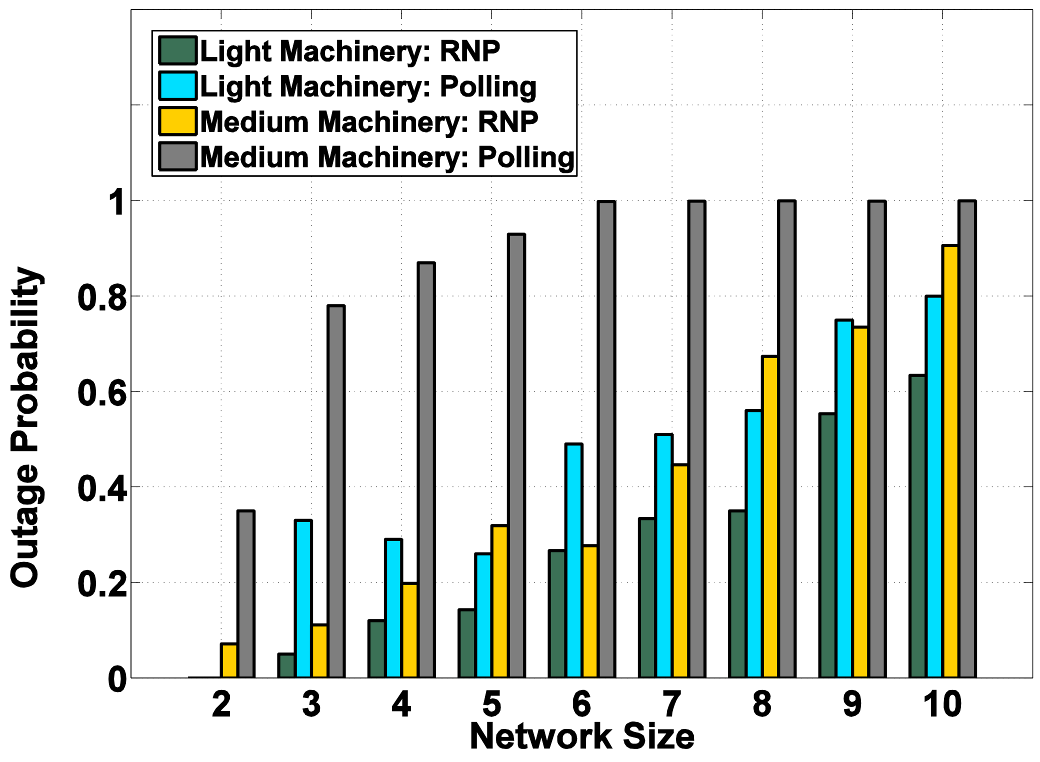


Figure 5.1.5. Outage probability due to unsuccessful attempts.

5.2 The UWASA Node

5.2.1 Introduction

Employment of Wireless Sensor and Actuator Networks (WSANs) for industrial automation provides cost effective replacement of cables, and easier and faster deployment compared to wired counterparts. WSAN allows development of reconfigurable and expandable systems as well as improving pertinency to applications which are impossible for cabled systems [12]. The cost effective replacement of cabled systems allows deployment of large number of measurement points, enabling development of self-organized, inherent colloborative processing and self-healing systems.

The wireless automation applications must confront with the problems associated with wireless communication, data processing and energy resource constraints of the nodes. The former problem is related to the intrinsic properties of RF waves and their interactions with the surroundings, which is addressed by physical layer of the communication hardware. Thus, the automation applications should be adapted to minimize the wireless communication related effects like packet losses and nondeterministic delays. On the other hand, the processing and energy resources of the nodes can be changed arbitrarily to meet the application demands at the cost of hardware redesign for different applications. The hardware update is usually the most expensive part of the development process, which also demands adaptations and major updates in the software.

In general, the node hardware is composed of a processor with low computational power and low capacity memory, and a short range, low bandwidth wireless transceiver. This architecture enables operation for a long time span with a limited energy supply (e.g. battery). The constraints in energy and data processing resources compel the WSAN deployments to sparse periodic measurements/control and/or event detections. However, the industrial automation applications are covered by a broader classification; the applications demanding dense measurements and fast event detections are quite common. Moreover, some wireless automation deployments may require some nodes running local data fusion, data compression, data aggregation, or control algorithms, while some other nodes are running only communication related tasks such as routing, relaying etc. Hence, even for a single deployment scenario of a wireless automation application may need different hardware platforms, which increases the development costs.

The wireless automation applications require reconsideration of tradeoff between power consumption and processing resources selected during the hardware design of conventional WSAN nodes. Usually, it is desirable to have a hardware platform which has adjustable tradeoff point, that is, the processing resources can be adaptively (in software) or easily (by stacking additional low complexity hardware) increased at the cost of increased power consumption. The current microcontroller technology, power management circuits, and System-on-Chip (SoC) wireless communication circuits make it possible to develop such a small form factor, easily upgradable WSAN node hardware. The basic node must intrinsically provide wireless communication hardware, and processing and energy resources extendibility interfaces in hardware and adaptation interfaces in software. Consequently, for each application, the basic node would remain the same, and the application dependent hardware demands would be fulfilled by low complexity slave modules, requiring straight forward software modifications.

In this work, we are introducing a modular and stackable wireless sensor platform, the UWASA Node [13, 14], which is a generic platform for wireless automation applications. The UWASA Node provides adaptation and extension interfaces both for software and hardware to enable reuse of the same platform in different deployments.

5.2.2 Stackable Hardware Platforms

The UWASA Node is designed to support broad class of applications ranging from low-power single processor wireless sensor node, up to applications with demands that can be met by multiple processors stacked on top of one another. Although stackable hardware platforms, such as industrial computer standard platform PC/104, are available for broad class of control and robotics applications, multiprocessor architectures are traditionally ignored by WSAN community. The complex power management, the complex timing management, interprocessor bus development, and the abstraction software development requirements of such stackable platforms make them impractical for WSAN deployments. On the other hand, high variation in resource demands of wireless automation applications motivates reconsideration of stackable platforms as WSAN nodes. However, the stackable hardware WSAN node can only be developed by effectively solving the engineering design challenges to maintain the advantages of WSAN deployments.

The stringent power constraints on WSAN nodes require reconsideration of stackable hardware platforms for fast and efficient organization of available resources to meet application demands. The power constraint of portable devices has motivated development of integrated Power Management Unit (PMU), which has extremely low quiescent current consumption and efficiency as high as 96%, while providing software controllable logic interfaces. Arbitrary number of such PMUs can be utilized to provide software controllable power management for stackable platforms. Likewise, the battery monitor and supervision needs of portable devices has driven to introduction of integrated Battery Supervisory Unit (BSU), which can be utilized for efficient dynamic power path management, battery charging, and charge level gauging. Consequently, recent technological advances in power management sub-systems for battery powered portable devices allow development of small form factor, software controllable power modules.

Placement of PMUs and BSUs in a software controllable separate hardware module provides easy way to adapt power demands of different applications. However, the instantaneous power consumption of microcontrollers is proportional to operating frequency, the types and the number of activated peripherals at a given time. Usually, applications require on-demand activation of some of the peripherals and microcontrollers to handle specific events. However, if these on-demand activated peripherals are integral to microcontrollers, the sequence for activation –

deactivation of peripheral and processor must be coordinated and tracked by an external logic. Therefore, power demands of an arbitrary application can only be minimized by duty cycling of processors and associated peripherals in a coordinated fashion, which requires software for tracking and updating the operating (power) state of individual microprocessors as well as associated power greedy peripherals.

It is very well known that the time reports of two software clocks driven by independent crystal oscillators deviate from each other due to non-ideal behavior of oscillators. The processors, which have independent oscillators, tend to have different notion of time, causing synchronization and calibration problems inside the hardware stack. In addition, the wireless automation applications demand coordinated operation within the network, requiring synchronized operation both locally in a single node and globally in the network. Consequently, the time management in the node among multiple processors must be provided in accordance with the global time of the network.

The most expensive development stage of WSAN applications is the embedded system development, which is composed of both embedded software and hardware developments. In this respect, although the stackable hardware decreases the costs associated with the hardware development, the embedded software development costs is linear with the number of processors, unless mechanisms for software reuse are defined. Usually, operating systems provide an abstraction of the hardware, allowing software reuse in different processor targets. However, the available operating systems for WSAN nodes are simple kernels providing only limited application programming interfaces. On the other hand, the microprocessors that can be utilized in the WSAN node hardware stack vary drastically from 8-bit Harvard architecture processors to digital signal processors with superscalar architecture. Such an irregular distribution of processor candidates prohibits limitation on a single type of operating system. Consequently, stackable hardware platforms for WSAN require definition of operating system independent processor abstraction software to allow software reuse with small configuration updates.

For single microcontroller architectures, the application dependent external devices (such as sensors) can only be attached to the system using the available peripherals. However, for stackable multiprocessor architectures the types and the number of peripherals changes as the microcontrollers in the stack are altered. Moreover, the specific peripheral interface to interconnect microprocessors in the stack may vary with required data exchange rate, prohibiting single bus definition. Such versatility causes development of different embedded software for different applications unless the underlying hardware peripherals are abstracted in software. Consequently, stackable hardware architecture requires development of abstraction software that allows development of applications independent of underlying hardware peripherals used for interconnecting processor and/or external device to the node, similar to middleware commonly utilized for such a purpose.

To sum up, stackable hardware architecture for WSAN nodes needs to address power and time management interfaces both in software and hardware. Moreover, the associated embedded software must be carefully developed to provide suitable level of hardware abstraction for software reuse and easy application development.

5.2.3 *UWASA Node Hardware Architecture*

The modular and stackable hardware architecture of the UWASA Node is shown in Figure 5.2.1. This architecture allows easy adaptation of hardware to different applications by means of Slave Modules, while providing moderate level of computational power and memory, and wireless communication hardware intrinsically in Main Module. The power distribution and management in the node is maintained by means of Power Module.

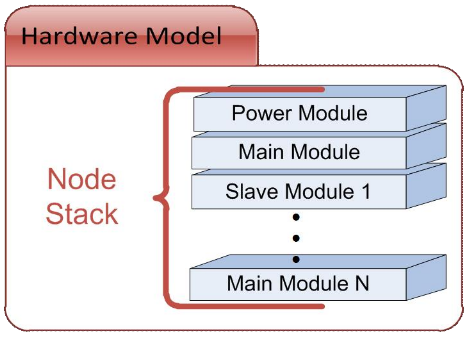


Figure 5.2.1. The hardware model of the UWASA Node.

The node is designed to have at least Power Module and Main Module, as illustrated in Figure 5.2.2. These two modules contain all intrinsic properties like wireless communication hardware, support for many peripheral interfaces, basic processing and memory, power management and distribution interfaces. The hardware stack contains some simple slave modules, in which the application dependent hardware is implemented. Regardless of the types and number of the modules in the hardware stack, the signaling and the power supplies are transferred between the modules by means of Hardware Stack Connector.

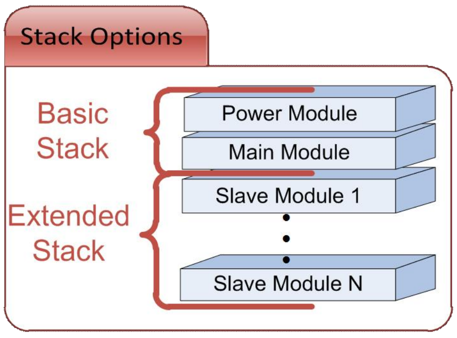


Figure 5.2.2. UWASA Node hardware stack options.

The node is designed to support a broad class of applications ranging from the ones demanding low-power, wireless transceiver only operation up to the applications demanding complicated interfacing and processing, as shown in Figure 5.2.3. The basic node can be operated as a simple wireless transceiver for low-power operation, or as a complete node with higher amount of resources.

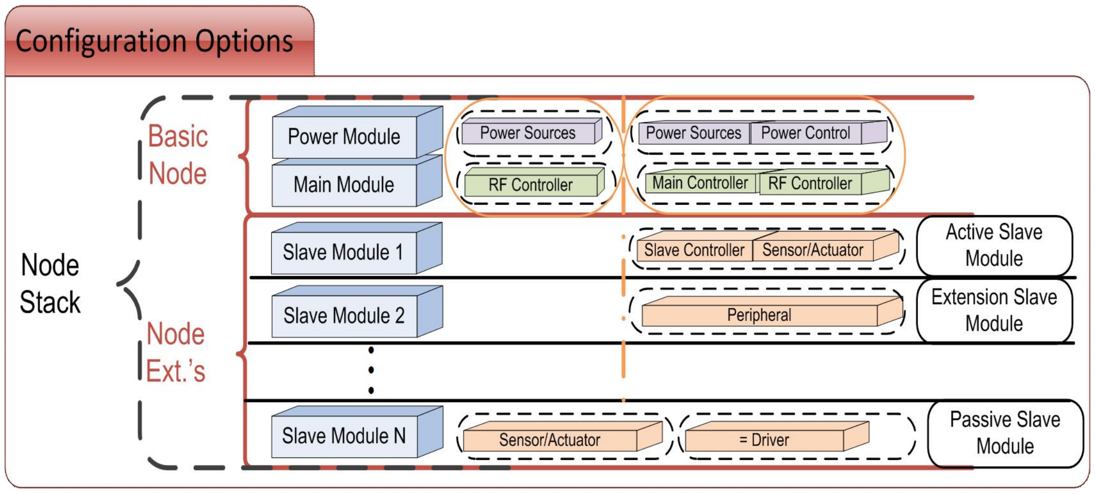


Figure 5.2.3. The possible hardware configurations of the UWASA Node.

5.2.4 *Power Distribution and Management*

The UWASA Node contains a dedicated Power Module for power supervision and management. The management interfaces of the Power Module are shown in Figure 5.2.4.

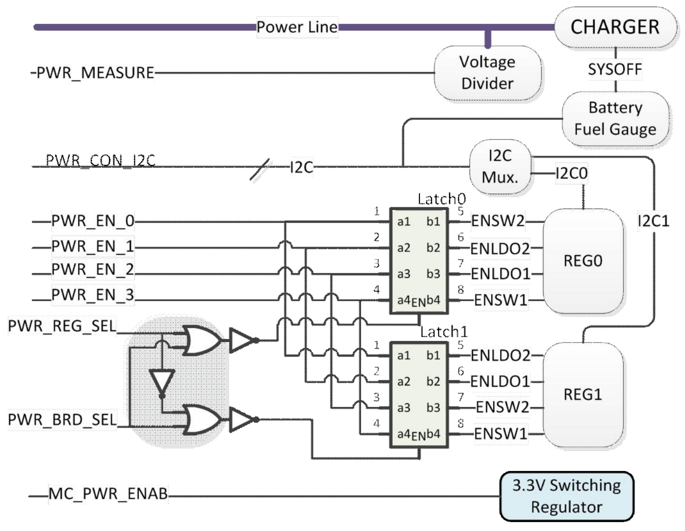


Figure 5.2.4. Power management interfaces.

The nodes can be deployed in environments that can provide multiple power sources. The UWASA Node supports two independent supplies, one of them being a rechargeable battery and the other one being any source meeting the requirements (preferably an infinite energy source). The UWASA Node is equipped with dynamic power path management hardware, which efficiently utilizes two supplies. If the infinite energy source can provide more power than the instantaneous demand of the node, the battery starts to be recharged.

The power module is equipped with 10 independent regulators. Two of these regulators have fixed voltage outputs to supply the main module. Other eight regulators have adjustable output voltage levels, and they can be disabled by logic controls. This feature allows adaptation of power resources according to demands of the slave modules.

The power module contains battery fuel gauge, which can be accessed by the main controller. This feature allows battery energy monitoring, and generation of accurate low power alerts. Moreover, this gauge can be used for quantification of power demand of application tasks, which is crucial for determination of energy demand of a specific application.

5.2.5 *Time Synchronization and Management*

For proper data acquisition and algorithm operation, a node has to guarantee synchronized operation between its processing units. Moreover, the complete node must comply with the network level time synchronization demands of the application. The in-node time management is performed by a heart-beat signal provided by the Main Module. This signal is used for time stamping of each data chunk that will be exchanged inside the node. Being simple, yet, this mechanism provides additional logic free time distribution between the processing units in the node. The network level time synchronization is handled by the Main Module. The output of the network level time synchronization algorithm compensates for the clock skew of the local clock, which in turn updates the heart-beat signal period. The time management support of the node is shown in Figure 5.2.5.

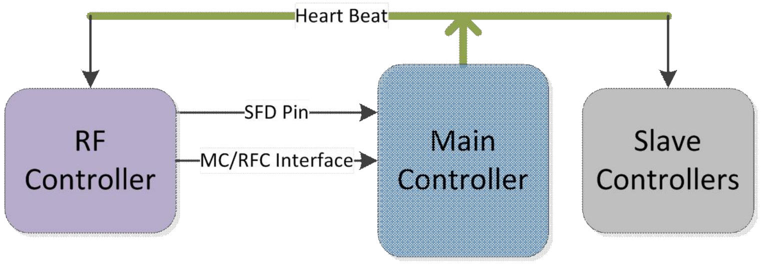


Figure 5.2.5. The time synchronization interfaces.

5.2.6 Hardware Modules

Main Module

The Main Module is the master module, and contains two processing units to allow transparent wireless communication while providing necessary means for stackable hardware architecture. The basic hardware blocks of the Main Module are shown in Figure 5.2.6.

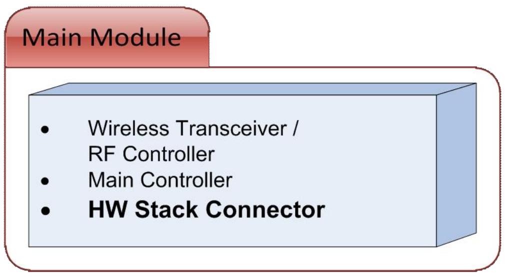


Figure 5.2.6. The basic hardware blocks of the Main Module.

The Main Module is responsible for wireless communication, managing in-node data exchange, performing data processing and decision making, while supervising power interfaces and managing in-node power mode. The component blocks of the main module are shown in Figure 5.2.7.

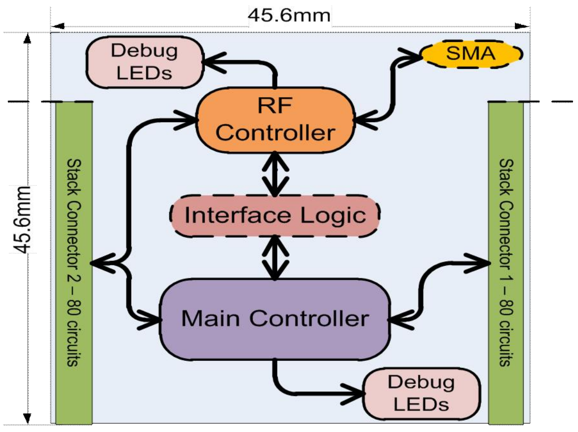


Figure 5.2.7. The components of the Main Module.

The main features of the main module can be summarized as follows:

- RF Controller and Main Controller are both programmable microcontrollers; RF Controller has 8051 based 8-bit processor that runs at either 16 MHz or 32 MHz, whereas Main Controller has ARM7TDMI-S 32-bit processor that can run at up to 72 MHz.
- RF Controller has integrated IEEE802.15.4 MAC, integrated positioning engine; whereas Main Controller has integrated Ethernet MAC, and USB2.0 Full Speed device, as well as many standard serial interfaces.
- RF Controller can switch off the power of Main Controller, when excessive process power is not needed; whereas, the Main Controller can control enable states and voltage level of power sources through the interfaces provided by the Power Module.
- Individual peripherals of Main Controller and RF front end of RF controller can be disabled.
- Operating frequencies of both RF Controller and Main Controller can be adjusted.
- Time synchronization is solved by the means of heart-beat signaling provided by either RF Controller or Main Controller. Network level time synchronization is ensured by the RF Controller.

Power Module

Power Module is responsible for providing power demands to all node modules. The basic hardware blocks of the Power Module are shown in Figure 5.2.8.

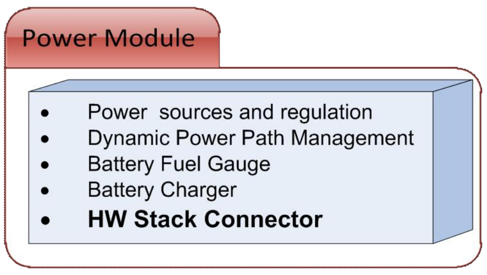


Figure 5.2.8. The basic hardware blocks of the Power Module.

This module contains power source path management, battery monitor, battery charger, voltage regulators, and power control logic block. Those components are shown in Figure 5.2.9.

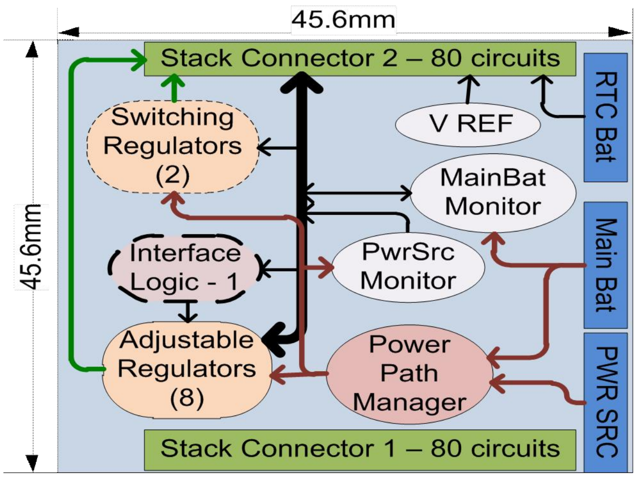


Figure 5.2.9. The components of the Power Module.

The main features of the Power module can be summarized as follows:

- ➢ The node supports LiIon – LiPo battery connection. The power module is designed to operate within the safety limits of LiIon – LiPo batteries.
- ➢ The power module contains integrated LiIon – LiPo battery charger, which allows on-line charging the battery when the other source is connected to a supply.
- ➢ Two independent supply options are provided through Power Module; the whole node supports dynamic power path management when the node is supplied via battery and other source (e.g. solar panel, vibration based power generator, USB connection, etc.).
- ➢ Instantaneous (power drain and charge) status of the battery can be monitored.
- ➢ The voltage level on the main power line of the node can be monitored.
- ➢ 10 independent power regulators are provided on the Power Module. 8 of them with adjustable output voltage levels; where 4 of them are switching regulators, and other 4 being linear regulators. Those regulators should be used to supply slave modules, to allow main controller to turn on/off whole slave module hardware, to minimize quiescent currents.

Slave Modules

Three types of Slave Modules are defined depending on the purpose:

Active Slave Module: This type of Slave Module has its own processing unit. Consequently, data processing – driver related issues are handled on the slave module itself.

Passive Slave Module: This type of Slave Module does not contain its own processing unit, but has driver related hardware of Sensor/Actuator, which is connected to one of the interface of the Main Module on the Hardware Stack Connector.

Extension Slave Module: This type of Slave Module is an extension to Main Module. For example, SRAM connected to External Memory Controller interface of the Main Module is this type of Slave Module.

The Slave Modules are intended to be the hardware implementations of the applications. The basic hardware blocks of three types of slave modules are shown in Figure 5.2.10.

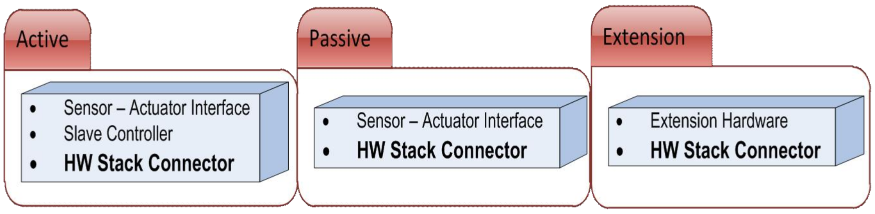


Figure 5.2.10. The basic hardware blocks of different Slave Module types.

Development Board

In order to simplify the development stage, the Development Board has been designed. This board aims to provide essential development interfaces for both RF Controller and Main Controller of the Main Module. The content of this board is shown in Figure 5.2.11.

In addition to debug emulators for both of the Main Module microcontrollers, the Development Board contains four buttons, connected to two External Interrupt pins of each microcontrollers, and UART-to-USB converters. These simple interfaces enable the developer to trace the state of the developed code.

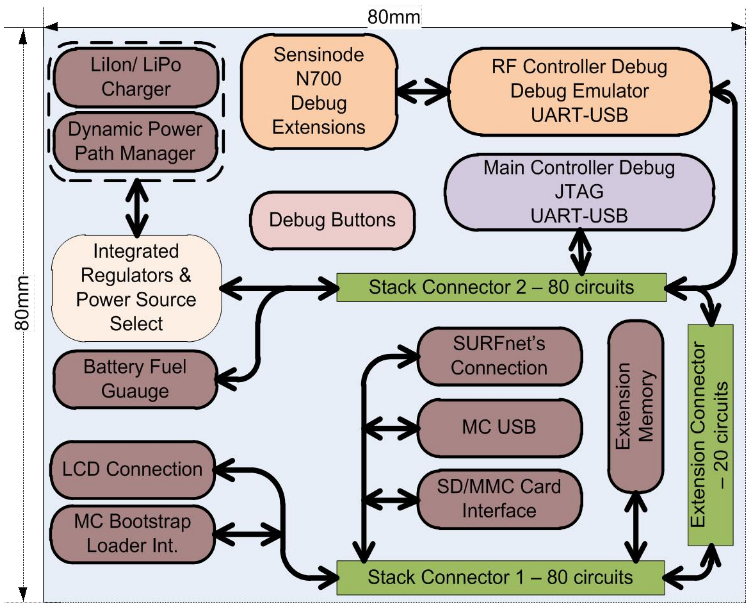

80mm
80mm
LiIon/ LiPo Charger
Dynamic Power Path Manager
Sensinode N700 Debug Extensions
RF Controller Debug Debug Emulator UART-USB
Main Controller Debug JTAG UART-USB
Debug Buttons
Integrated Regulators & Power Source Select
Stack Connector 2 – 80 circuits
Battery Fuel Guauge
SURFnet's Connection
MC USB
SD/MMC Card Interface
Extension Memory
Extension Connector – 20 circuits
LCD Connection
MC Bootstrap Loader Int.
Stack Connector 1 – 80 circuits

Figure 5.2.11. The functional blocks of the Development Board.

5.2.7 Software Architecture

The stackable hardware architecture requires a higher level of hardware abstraction in the software to reach software reuse in different deployments. For this purpose the UWASA Node software is designed to have multiple levels to reach the required level of hardware abstraction. The UWASA Node software is based on the modules depicted in Figure 5.2.12.

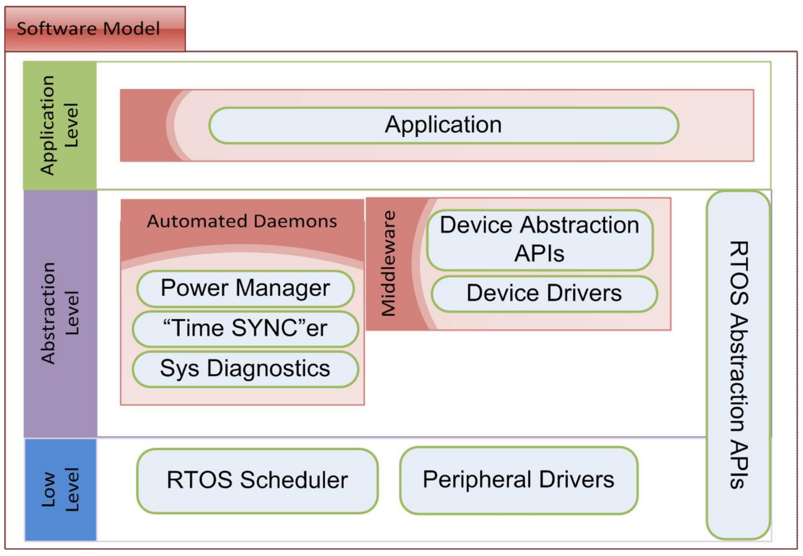


Figure 5.2.12. Software Model.

The required level of abstraction is reached in three layers. The first layer is low-level software associated with the underlying hardware target. It is composed of peripheral drivers and Real-time Operating System (RTOS) scheduler which both contain hardware target related components. The second layer is composed of automated daemons and middleware to reach abstraction from both RTOS and Peripherals. The last layer is the Application layer containing only the application software.

RTOS Abstraction

In this software model, by RTOS we do not mean a specific Operating System (OS), but a software component providing multi-tasking and synchronization interfaces. Thus, the software model only assumes existence of an OS with Signaling and Mutual Exclusion APIs (both are part of inter-task synchronization), not a particular type. Such an approach is only possible in case the minimum required programming interfaces of any candidate RTOS APIs are carefully selected and kept as low as possible.

Low level software layer require synchronization interfaces to indicate occurrence of specific types of events to upper layers. Depending on the type of the peripheral, these events may be physical instantaneous events (e.g. voltage level change), or software events (e.g. memory transfer complete); usually accompanied with additional information such as occurrence time or status. Hence, the hardware abstraction requires presence of event signaling API.

The multi-task application development provides easy application development at the cost of increased software complexity. The software resources such as software or hardware resources like I2C interfaces require access guards to allow intervention free operation since the application tasks may request to update the state of a resource before the former task finishes its request. For this purpose, every shared resource must contain a mutual exclusion mechanism as access guard.

It is quite common to provide inter-task data exchange mechanisms, like queues, as a part of RTOS API. These mechanisms utilize signaling API to allow blocked access to specified memory region. On the other hand, it is a straight forward task to implement guarded memory buffer structures using either mutual exclusion or event signaling APIs. Thus, we do not assume presence of such a data exchange mechanism, but provide two mechanisms which are described next.

All the software components in Figure 5.2.12 make use of two basic programming interfaces: the generic buffers and the buffer of buffers.

Generic Buffer API

The Generic Buffer API is an efficient, data guarded implementation of Generic Buffer data structure. A Generic Buffer is defined as a structure which contains a data array, a head pointer and a tail pointer. The data array is the location of the buffer memory. The head pointer is the memory location where the upcoming data will be stored. The tail pointer is the memory location where the first data element has been stored. These two pointers only guarantee linear FIFO operation. In order to obtain circularity, the data structure is augmented with buffer length constant field. A buffer data structure is visualized in Figure 5.2.13.

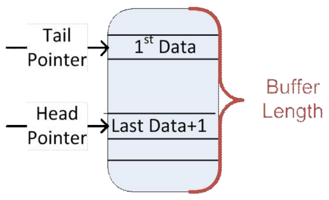


Figure 5.2.13. Generic Buffer Structure.

For most of the applications of buffers, it is necessary to inform consumer task whenever the producer task requests to do so. Such a blocking access can easily be used to awake consumer tasks in case the buffer if filled up to a threshold specified by the producer task. This mechanism is roughly visualized in Figure 5.2.14.

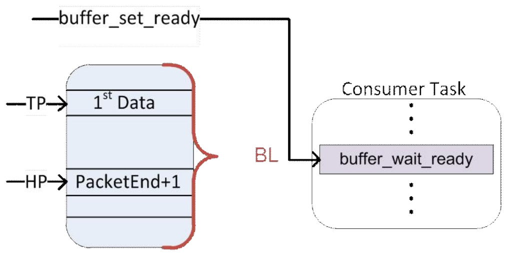


Figure 5.2.14. FIFO Buffer consumer signaling.

Buffer of Buffers API

The Buffer of Buffers (BufBuffer) API is an efficient interface to circular (ring) buffer of FIFO Buffers. As the definition implies, it is a fully circular buffer, but the elements are pointers to FIFO Buffers, as depicted in Figure 5.2.15.

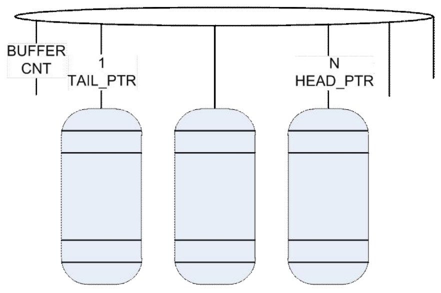


Figure 5.2.15. BufBuffer Data Structure.

The motivation behind such a mechanism is to allow producer to fill a buffer independent of the consumer task. This API signals the consumer about availability of ready buffer, while producer continues to fill the next empty buffer in the circular queue. At that point, the producer task is not concerned about consumption of previous buffer, and continues its operation. The consumer task, however, reads or manipulates data from the ready FIFO buffers. After completing the consumption, it goes to blocked state to wait for the next buffer to be ready.

RTOS Dependence

In the current software architecture, the RTOS is only required for resource guarding and event signaling. If the application does not require multi-tasking, the software architecture does not importunes on having OS scheduler, but only synchronization APIs. Thus, the presence of actual RTOS is not critical as long as these two mechanisms are provided by some other means.

Low Level Software

The low level software is composed of peripheral drivers and RTOS Scheduler. In this level, the hardware related peripheral initialization and configuration is performed. In addition, it provides suitable programming interfaces for abstraction layer.

The RTOS scheduler, being partly related to hardware, is considered in this level of the software architecture due to the dependence of higher layer software on it. The multi-tasking and inter-task synchronization APIs, accompanying to RTOS kernel, are used by the peripheral drivers to signal the upper layers whenever a specific event occurs. The peripheral drivers provide necessary level of abstraction to allow actual target processor independent development in higher layers. For example, the UART peripheral driver of a specific microcontroller contains all software development of hardware dependent parts, and provides specified interfaces for device drivers.

Abstraction Level Software

The Abstraction Level software contains automated daemons and middleware. The relations between software components of this layer are visualized in Figure 5.2.16.

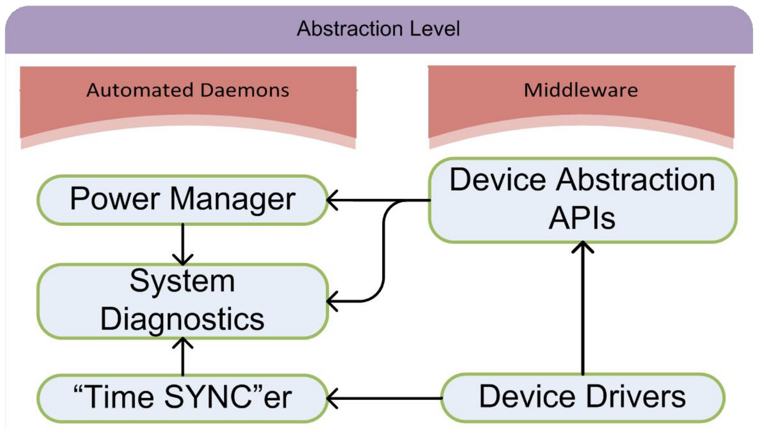


Figure 5.2.16. The relations between Abstraction Level Software components.

The automated daemons perform periodic tasks such as time synchronization, system status monitoring, and power state monitoring. Since the normal application execution does not require explicit interaction with these tasks, they run in the background to maintain proper operation conditions for applications. It should be noted that these daemons do use the middleware to interact with external devices in the stack.

The middleware is composed of device drivers and device abstraction APIs. The device drivers are the software components that contain necessary software interfaces to utilize external application dependent devices attached to the node. The low level software of microcontroller peripherals are utilized by device drivers to reach target platform independent driver development as depicted in Figure 5.2.17.

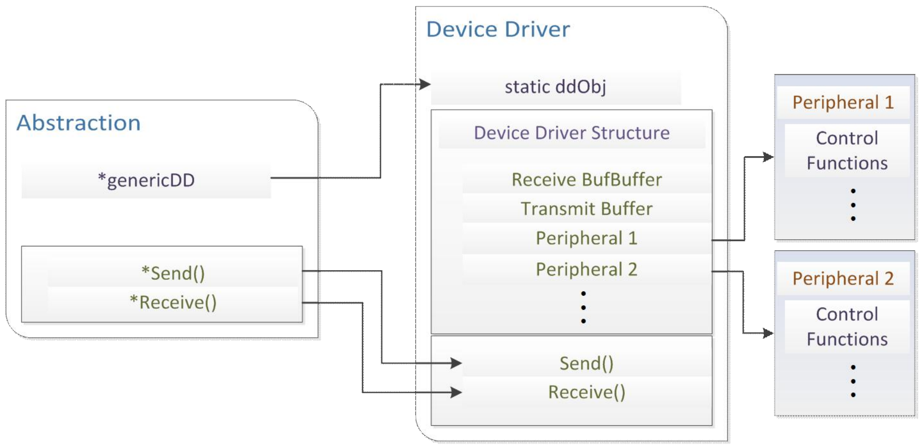


Figure 5.2.17. The device abstraction.

One device driver may use multiple peripheral drivers. However, such a detail is not usually a concern for the higher layers since they only depend on the information received or send from the device. Consequently, it is possible to define an abstracted, generic, device driver which resembles all possible types of devices.

The multiprocessor stackable architecture requires a definition of data exchange mechanism among the microprocessors. Based on the abstracted device drivers, the inter-processor communication is illustrated in Figure 5.2.18.

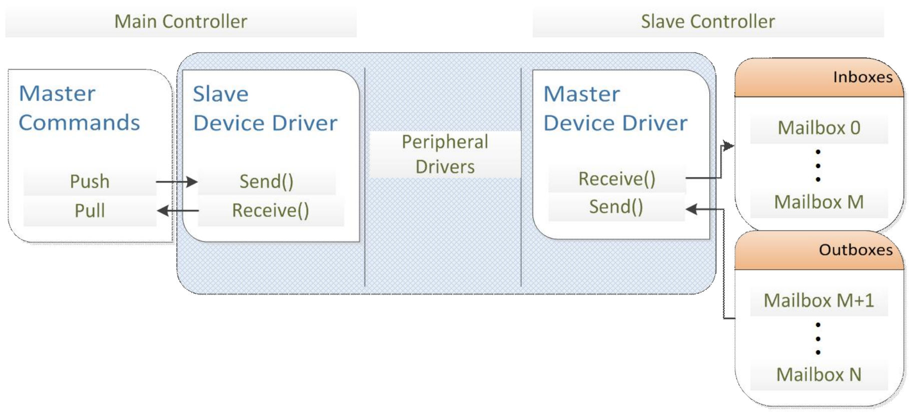


Figure 5.2.18. The inter-process communication mechanism.

The introduced inter-processor communication mechanism is based on access to shared memory located in the slave microprocessor. Such an approach can be utilized by the application developers to provide unified access to all types of slave modules, since the application layer only assumes push or pull type of data exchange, not a specific type of device driver or peripheral driver.

Application Level Software

The Application Level software only contains application tasks which utilize the Abstraction Layer software for easy development. Although the number of tasks is application dependent, for most of the cases, the number is limited from 1 to 3.

5.3 SurfNet

SurfNet (Surf technology - **S**einäjoki **U**AS **RF**) is a sensor network architecture consisting of two different hardware platforms, network protocol and a programming environment for a low-power, low-size wireless sensor nodes [15]. It is developed in the School of Technology in Seinäjoki University of Applied Sciences (UAS). The development done in GENSEN-project continues the earlier research of wireless technology in Seinäjoki UAS.

5.3.1 SurfNet Platform v 1.0

According to our part of the research plan, we assembled a series of small-size wireless nodes for developing and testing purposes [16]. The node includes nRF24LE1 radio controller and MMA7360 3-D acceleration sensor. The size of the node is 23 x 14 x 5 mm. We developed also a USB-SPI bridge module for programming purpose and PC-based programming routines for radio controller which is using the USB-stick as illustrated in Figure 5.3.1 [17].

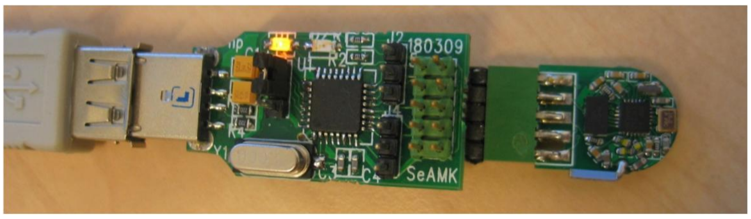
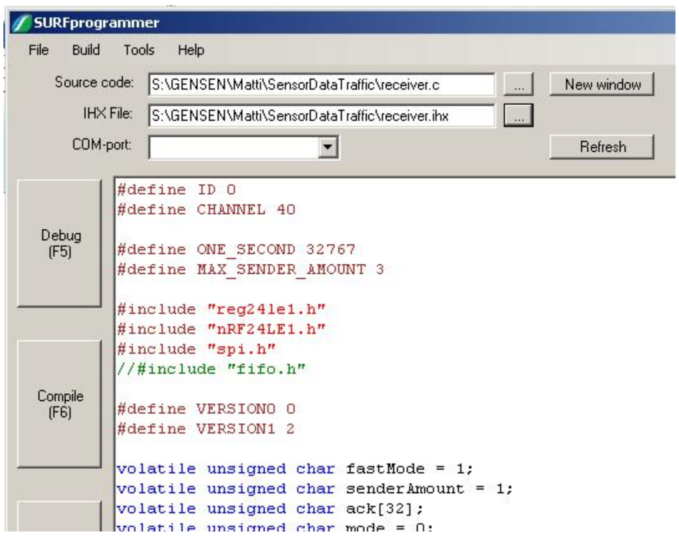


Figure 5.3.1. USB-SPI bridge with wireless node.

5.3.2 SurfProgrammer

A complete programming environment for wireless SurfNet nodes called SurfProgrammer was developed during the project. It includes a source code editor with C-language, open source SDCC C-compiler and transfer routines to and from radio controller program memory. The SurfProgrammer presented in Figure 5.3.2 includes an option to program the nodes over air interface. All source codes and documents are available at Seinäjoki UAS Embedded Systems website [15].



S:\GENSEN\Matti\SensorDataTraffic\receiver.c

S:\GENSEN\Matti\SensorDataTraffic\receiver.ihx

```
#define ID 0
#define CHANNEL 40

#define ONE_SECOND 32767
#define MAX_SENDER_AMOUNT 3

#include "reg24le1.h"
#include "nRF24LE1.h"
#include "spi.h"
//#include "fifo.h"

#define VERSION0 0
#define VERSION1 2

volatile unsigned char fastMode = 1;
volatile unsigned char senderAmount = 1;
volatile unsigned char ack[32];
volatile unsigned char mode = 0;
```

Figure 5.3.2. SurfProgrammer.

5.3.3 *SurfNet Mesh Network Topology*

A mesh networking protocol for SurfNet was also developed. The target was to have low-power nodes with multi-hop routing such that every node can act as a router. Applied routing method is so-called gossip routing, in which every node collects all most important measured data of all nodes in network. The goal was reached with minimum 0.25% RF duty ratio and 1.25% processor active duty ratio with 1.0 second synchronizing period. The mean supply current of one node is 0.035 mA in minimum, but in real application it is about 0.1 mA. The current curve during active 10 ms period inside 1.0 second synchronization period is shown in Figure 5.3.3 with the following time slots: 1) Controller starts by interrupt, 2) Crystal starts, 3) Upload message, 4) Receiving active, 5) Process received messages, 6) Sleep state, 7) Possible message send pulse. Also a positioning method is added into topology, in which every node uses neighborhood connections and fixed-position anchor nodes to estimate its own X-Y position [18].

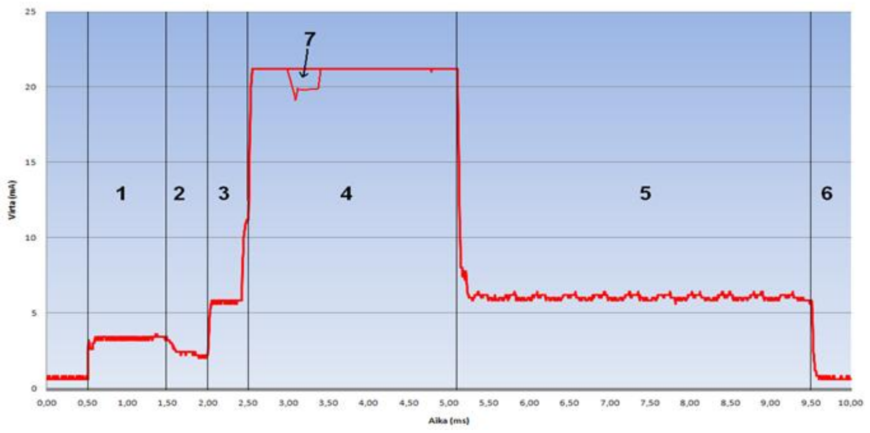


Figure 5.3.3. Active period current curve [18].

5.3.4 Bridging SurfNet and UWASA Node

We made a SurfNet Bridge Module to connect SurfNet nodes with UWASA Node. It operates as a sink node in network and communicates via SPI bus. A wireless bridge node collects the measured data from the network via gossip routing, as all other nodes. In our test cases we have connected the bridge module node with UWASA Node. It can download all data measured by SurfNet network, pass it to UWASANode and send commands from UWASA Node to SurfNet. In our research we use normally the same type of USB stick what we use for the programming of the nodes (Figure 5.3.1). In addition to programming features, it has a direct bridge or pipe via virtual COM port in PC to wireless bridge node in USB stick. New SurfNet network commands were also added by using the bridge module. Commands are transferred using flooding method. With these commands it is possible to set up the initialization features in SurfNet network: node ID’s, synchronization period, RF channel, RF power, subnet address and positioning features. In addition to sending commands it is also possible to set and read the application depending variables of every individual node.

5.3.5 Monitoring Software

The monitoring software collects both positioning data and measured sensor values of all nodes in the network via bridge module. The positioning is shown in graphical mode and as a table. The measured values are shown as node position point colors or background and foreground colors in the table. The colors are depending on the minimum or maximum limits of each sensor. The network command features are included to this monitoring software. This software was tested in the cattle house case (see Subchapter 6.5).

The monitoring software was also developed further for high-speed data flow. With this software version it is possible to collect data in a rate of 3000Hz, show the curves and save measured data in database. This software was used in the wind turbine case (see Subchapter 6.2).

5.3.6 SurfNet Platform V 2.0

The second revision of SurfNet platform was developed during the project to have more flexibility with the power source and sensor set. The new node platform is modular and stackable having several power supply options and several sensors. The alternative power supplies are battery, solar panel, peltier element and magnetic generator. The alternative sensors measure temperature, humidity, acceleration, light and voice. The size of the node is 25 x 16 x 8.5 mm added with alternative modules (Figure 5.3.4).

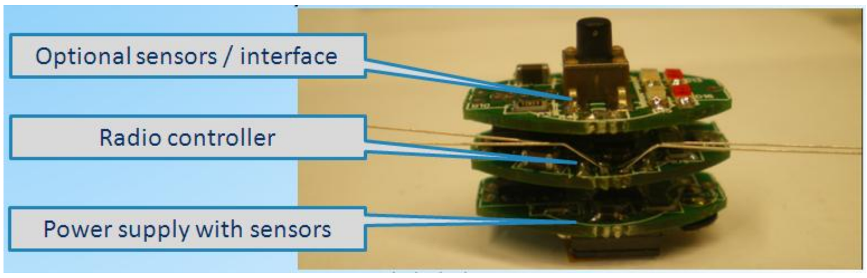


Figure 5.3.4. SurfNet stackable platform.

6 TEST CASES

6.1 Industrial Environment

6.1.1 The Radio Environment

The coexistence of high power wideband wireless local area networks (WLAN) and low power wireless sensor and actuator networks (WSANs) is challenging. In the worst case, when the channel occupancy of the WLAN is high, there could be only one channel available for IEEE 802.15.4 sensor network operation. Hence the motivation to utilize time and antenna diversity in sensor network is to mitigate the effects of fading and thus improving the reliability of the system while operating on a single channel. Antenna diversity technique consists of using multiple antennas on the sensor nodes. If the antennas are at least half a wavelength ($\lambda/2$) apart from each other, and there is enough scattering in the environment, then each antenna would see independent fading. On the other hand, since the packet transmission rate in sensor network is low, antenna switching could still be useful when combined with automatic retransmissions. If retransmission takes place immediately once the time window expires, then the time between consecutive transmissions is still short and antenna switching would be useful in providing diversity gain. That is, minimizing the correlation of the signal-to-noise ratio (SNR) between the consecutive packets. Therefore, we have performed some pre-study experimental analysis to evaluate gains for time and antenna diversity in wireless sensor networks.

6.1.2 Laboratory Scale Experiments

As a general evaluation, we have modeled uncorrelated multipath fading channels using a well-known tool PROPSim C2 [19]. This powerful channel emulator gives more accurate control of the emulated wireless links. Developed by Elektrobit, PROPSim provides 30 MHz RF bandwidth and 24 fading channels with a 0.1ns delay resolution, thus allowing to model link quality to be different for a combination of transmitter and receiver antenna pairs. The measurement setup is illustrated in Figure 6.1.1. We have used preloaded channel settings for a multi-tap indoor commercial environment.

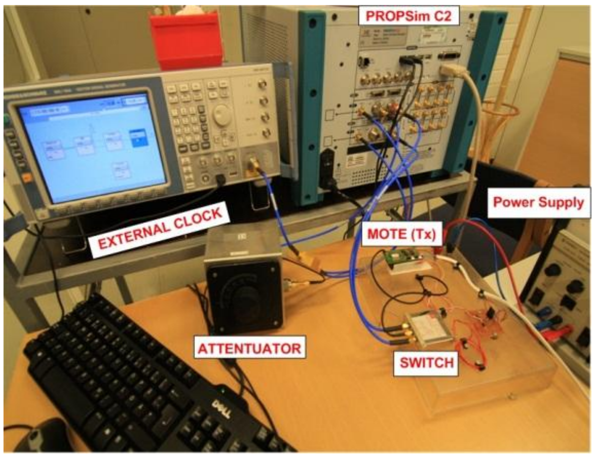


Figure 6.1.1. Simulation setup with PROPSim C2.

The results can be summarized as follows:

1. Using receiver selection diversity our results indicate that the packet delivery ratio can be increased to a maximum of 50% compared to using single antenna system.

2. Antenna switching gave us 70% packet delivery ratio compared to 40% using single antenna system.

6.1.3 Experiments in Industrial Environment

Tests were also conducted in typical industrial environment, which includes metal structures. The setup is shown in Figure 6.1.2. We developed a dual antenna system which enabled fast switching between two antennas. At the receiver end, data from the transmitter is collected by four nodes. Since there are 4 links available for each transmission on Antenna 1, the packet experiences four different channel characteristics and with fast re-transmission on Antenna 2 we achieve time and antenna diversity. Several measurement campaigns were done, with both 1s and 0.1s packet intervals.

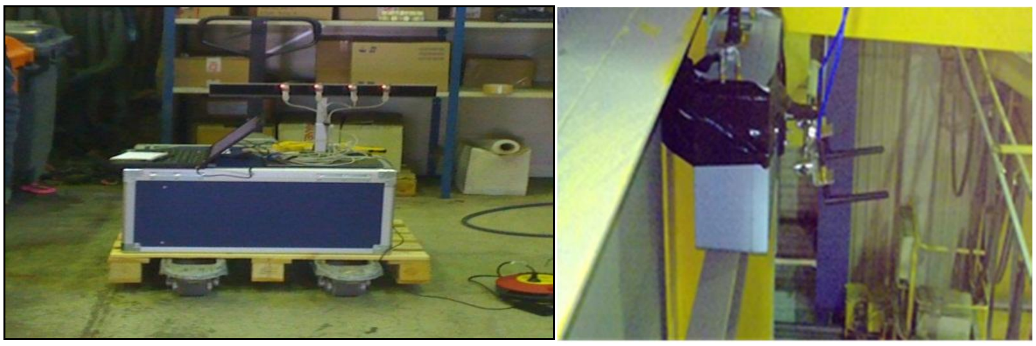


Figure 6.1.2. Industrial hall Tests a) Dual Antenna System (Transmitter) b) Receiver c) Packet Delivery Ratio.

A common way to model a network with packet drops is the Gilbert-Elliott (G-E) model, which is based on the Markov-chain. The G-E model has two states: one corresponding to good (G) and the other to bad (B) conditions, with separate packet drop probabilities in the good and bad state. The transitions between the states follow a two-state Markov model.

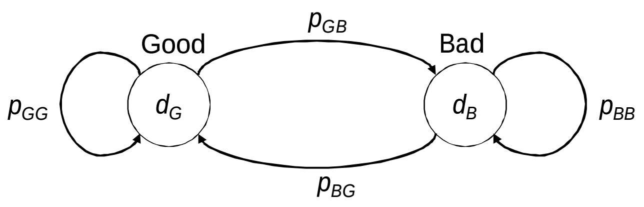


Figure 6.1.3. Gilbert-Elliot model with states Good and Bad. State-transitions and probabilities indicated.

Realistic packet drop models of wireless networks are needed to study WNCSs, since information loss ought to affect the control performance. One approach is to measure the packet drop with wireless nodes in authentic environments and use these results to build data-based network models. The physical properties of an existing radio environment are assessed by carrying out actual measurements at the target site.

Measurements with the same antenna array equipment are performed in an industrial assembly hall and an office. Measurements are made in different parts of the hall, which can be categorized as light: open space, medium: mostly open with machines standing on the floor, and heavy: metal racks of tools obstructing the line-of-sight. The distances between the transmitter and receiver for the different measurements are in the range of 25 - 35 m. The office is an indoor environment with plaster walls.

The measured packet drop probabilities of the office and industrial hall are shown in Figure 6.1.4. The packet drop probability varies from location to location and there is a significant variation between the antenna pairs. This implies that the signal strength is very sensitive to the antenna location, due to multipath fading.

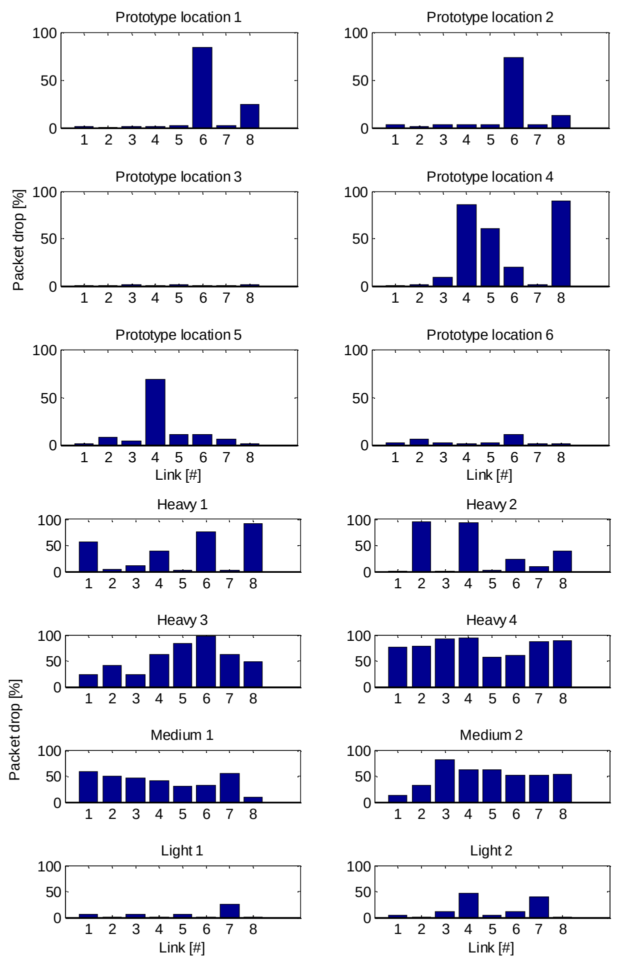


Figure 6.1.4. Measured packet drop probabilities in the office and in the industrial hall.

Based on the measurements performed in the industrial hall and the office, Gilbert-Elliott packet drop models are identified from the data. As an example, Figure 6.1.5 displays typical results. The G-E channel model is integrated to PiccSIM simulator and then used to evaluate various simulation based case studies in realistic channel conditions.

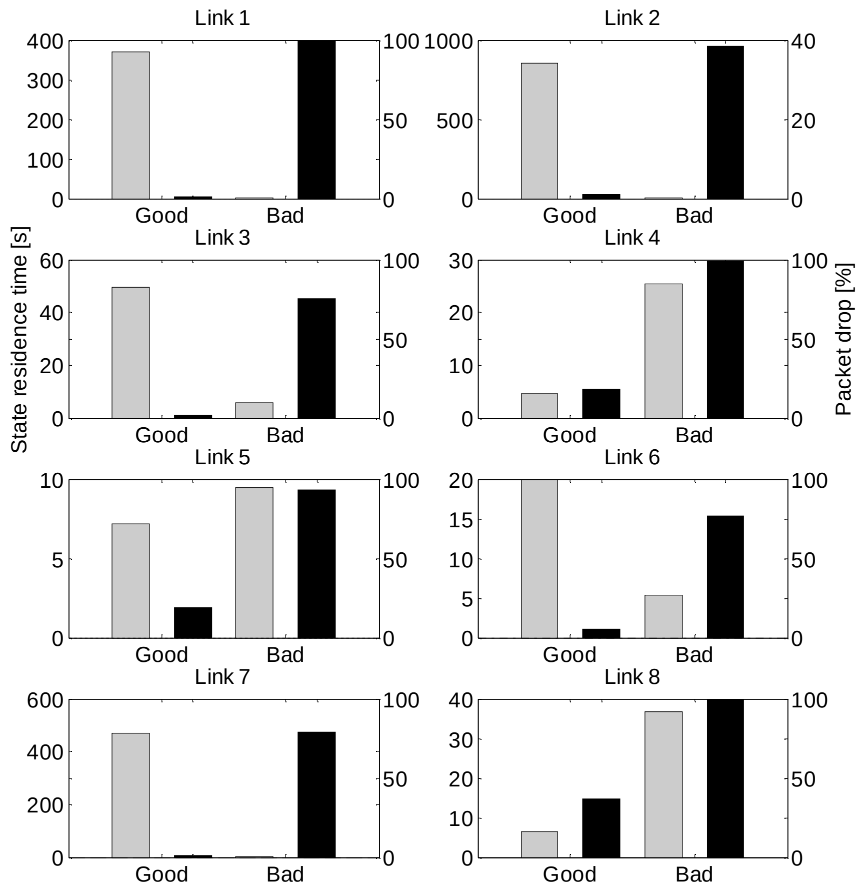


Figure 6.1.5. Gilbert-Elliott model for each link of prototype location 4 in the office. Grey bar indicates mean state-residence time and black bar packet drop probability.

6.1.4 Kick-Off Tests in Industrial Environment

In the first test, ten nodes equipped with a temperature and humidity sensor were deployed in a large industrial hall. Several shelves and other large objects, e.g. boxes, electric appliances, machines, etc. were found in the deployment area. After building a tree-like topology with Kick-Off, a simple communication schedule was defined and then distributed to the time synchronized nodes. Each node turned its radio on only in correspondence of its transmission and reception slots. The length of a TDMA slot was 200 ms. The collected data was transmitted towards the sink node along the routes defined by Kick-Off. Each node was required to transmit towards the sink node ten thousands data packets containing a temperature and humidity sample. The results of the deployment are presented in Figure 6.1.6.

In this first deployment, the low packet delivery ratios (PDRs) observed from the nodes topologically far away from the sink were caused by the presence of a software bug in the time synchronization software.

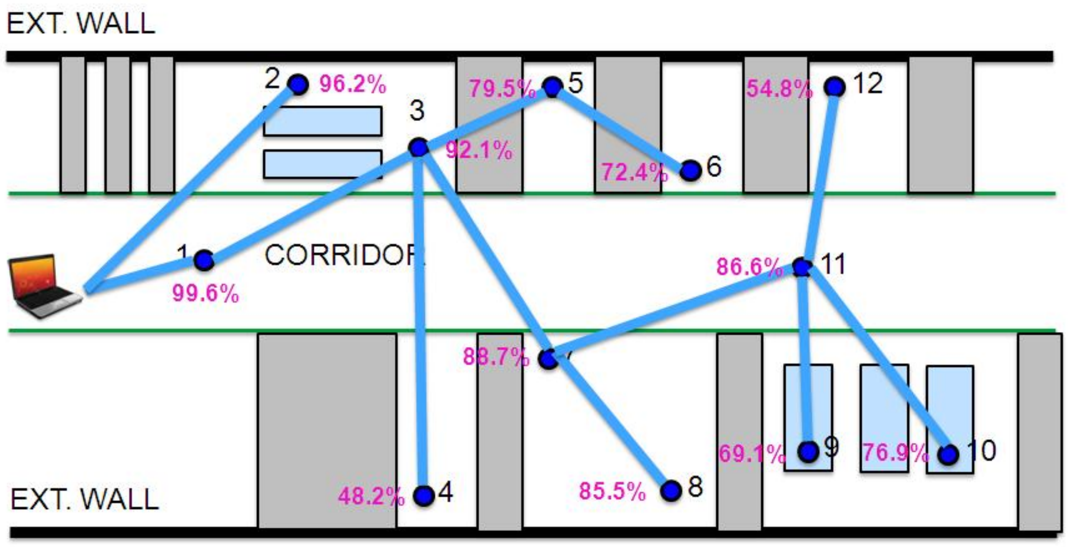


Figure 6.1.6. The tree-like topology defined by Kick-Off in the deployment carried out in the industrial hall. The percentages close to the nodes represent the recorded packet delivery ratios.

6.2 Wind Turbines

6.2.1 General

Modern wind turbines are remarkably bigger and heavier than the ones produced 20 years ago. For example, the length of the blade in Mervento’s 3.5 MW wind turbines is going to be 55 meters, which makes the turbine diameter to be 110 m and gives some idea about the height of the nacelle. As a consequence, there is an increasing interest to monitor vibrations and possible structural changes of the wind turbines. Frequency converters are used to produce AC and to connect the wind turbines, or wind turbine parks, to the external electric network. If sensor data is used in the control system and if wireless control loops are built by using WSNs, an interfacing between WSN and frequency converter is needed.

In GENSEN wind turbine case we verified the feasibility of our sensor system in blade vibration monitoring. We also interfaced the sensor network with Vacon frequency converter. Developed interfacing can also be used with other wireless automation applications where frequency converters and WSNs are used.

6.2.2 Blade Vibration Monitoring

Blade vibration monitoring was tested in a site of two wind turbines in Kauhava. These wind turbines are older and their nacelles are located only on 30m height, but the environment was good enough for the first proof of concept tests.

In the experiments, a wireless SurfNet node with 3D acceleration sensor was mounted on the tip of the wind turbine blade. A laptop computer with USB-SPI bridge and sink node collected real-time data, which was shown on the computer screen and written to the file. As a result of the challenging test setup, these first tests were done by using only one 3D acceleration sensor. The applied sensor was able to measure without saturation only up to 1.6 g acceleration. The test setup is shown in the Figure 6.2.1.

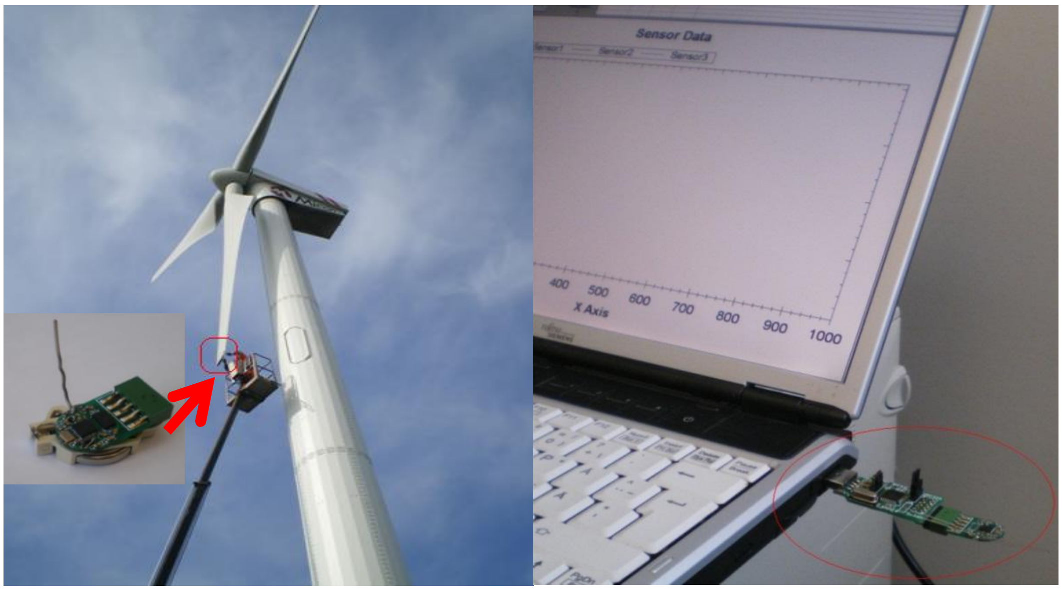


Figure 6.2.1. Blade vibration monitoring test setup.

There were no transmission errors when the blade was not moving, and the observed error rate (the amount of lost packets) was 0 % as shown in Figure 6.2.2. The distance between transmitting and receiving node was about 20 m.

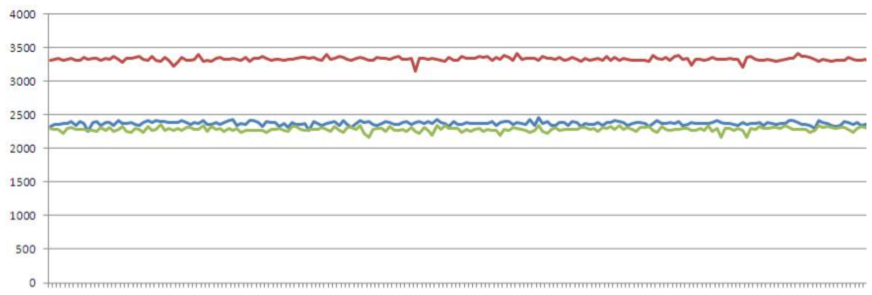


Figure 6.2.2. Measured acceleration data, while the turbine did not rotate.

Once the turbine stated to rotate, the 1.6 g acceleration limit was reached in vertical direction and the signal saturates as show by the red curve in Figure 6.2.3. There was also 33-36% packet loss existing in cycles every time when the blade equipped with the transmitting node was in the most remote location from the receiving node. This result addresses us to use multi-hop connection where one node acts as a router in the middle, when transmitting the data. The packet loss can be remarkably reduced in that way. Measurement data can also be buffered in the measuring node before the transmission such that it can be re-transmitted if the packet is not received for the first attempt.

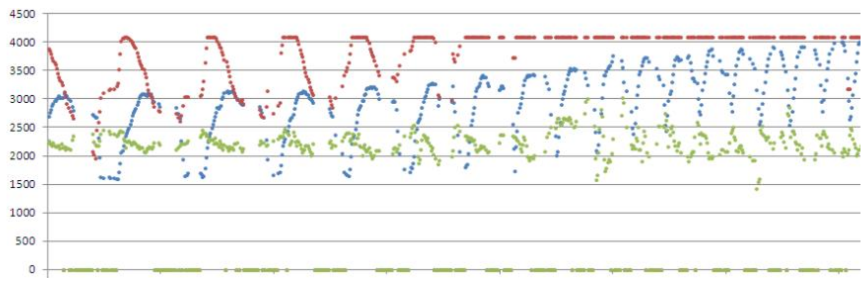


Figure 6.2.3. Measured data while the turbine starts to rotate.

6.2.3 *Interfacing WSN with Vacon Frequency Converter*

Vacon Frequency converter uses internal Modbus card (RTU, half-duplex). A slave module to communicate over RS485 has been designed to UWASA Node and the Modbus protocol has been implemented to the main controller of UWASA Node main module. The slave module has been designed such that it is isolated from the electromagnetic disturbances of the system.

Developed interfacing enables communication to both directions. Sensor data can be sent to the frequency converter and either read from converter's LCD screen or sent further by using frequency converter's Ethernet connection. The sensor data can also be used in the internal computation in the frequency converter to control it. On the other hand, it is also possible to send control commands from frequency converter to sensor network by using the same interface.

6.3 Distributed Energy Production

6.3.1 General

A suitable environment for distributed energy production monitoring and control demonstration case was provided by a museum and observatory Meteoriihi, which is located in the middle of Söderfjärden in the Southern side of Vaasa, see Figure 6.3.1. Söderfjärden is a meteor crater, which is nowdays a circular field area having a diameter of about three kilometers. Because of its uniqueness, the field area as well as the villages on its edges has a legal status of protected cultural environment, which makes it impossible to build electric cables or air wires to the field area. As a consequence, the electricity the Meteoriihi building needs must be produced locally. There is a small wind turbine, two solar panels and a diesel generator to provide the power supply. Diesel generator is also used in another research project to test how the generator run with different biodiesels and the on fly shifting from one diesel type to another.

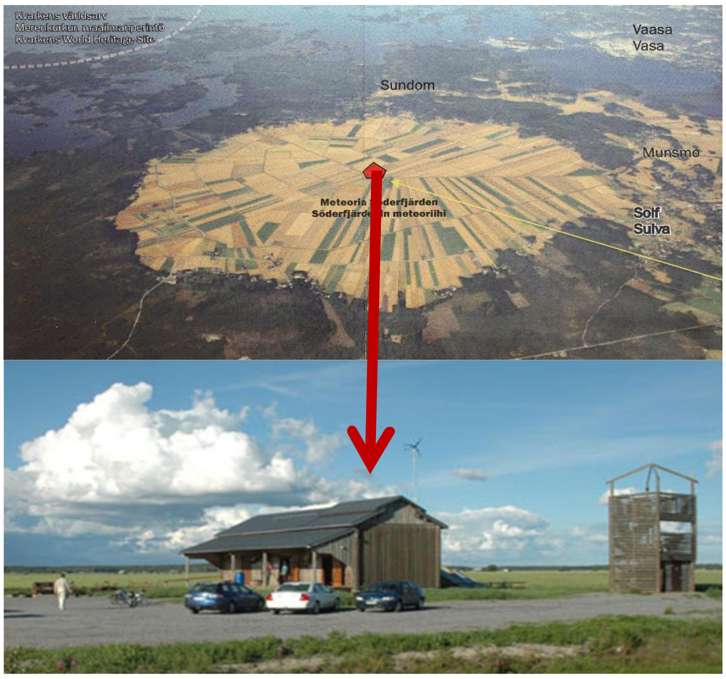

Kvarkens världsarv
Merenkurkun maailmanperintö
Kvarkens World Heritage Site
Vaasa
Vasa
Sundom
Munsmo
Meteoria öderfjärden
Söderfjärde in meteoriihi
Solf
Sulva

Figure 6.3.1. Museum and observatory Meteoriihi in the middle of Söderfjärden meteor crater.

In the context of GENSEN-project, a system to monitor and control the electricity production in Meteoriihi was created. A system consists of Vaisala WXT520 Weather Transmitter and wireless sensor network. It is connected to the University of Vaasa over long distance link which uses GPRS.

6.3.2 *Developed Architecture*

There is a need to measure the weather parameters, light intensity, temperature, humidity, battery bank charging level, the power produced by each of the sources and diesel generator burning gases. Vaisala WXT520 measures six weather parameters: temperature, humidity, air pressure, rain intensity, wind speed and wind direction. Light intensity, temperature and humidity inside the building, power production and burning gases must be measured by the sensors connected to the sensor nodes. In the first version of the architecture we applied Sensinode Nanoseries sensor nodes with Vaisala WXT520 Weather Transmitter and Telit TM862 GPRS modem [20]. System architecture is illustrated in Figure 6.3.2.

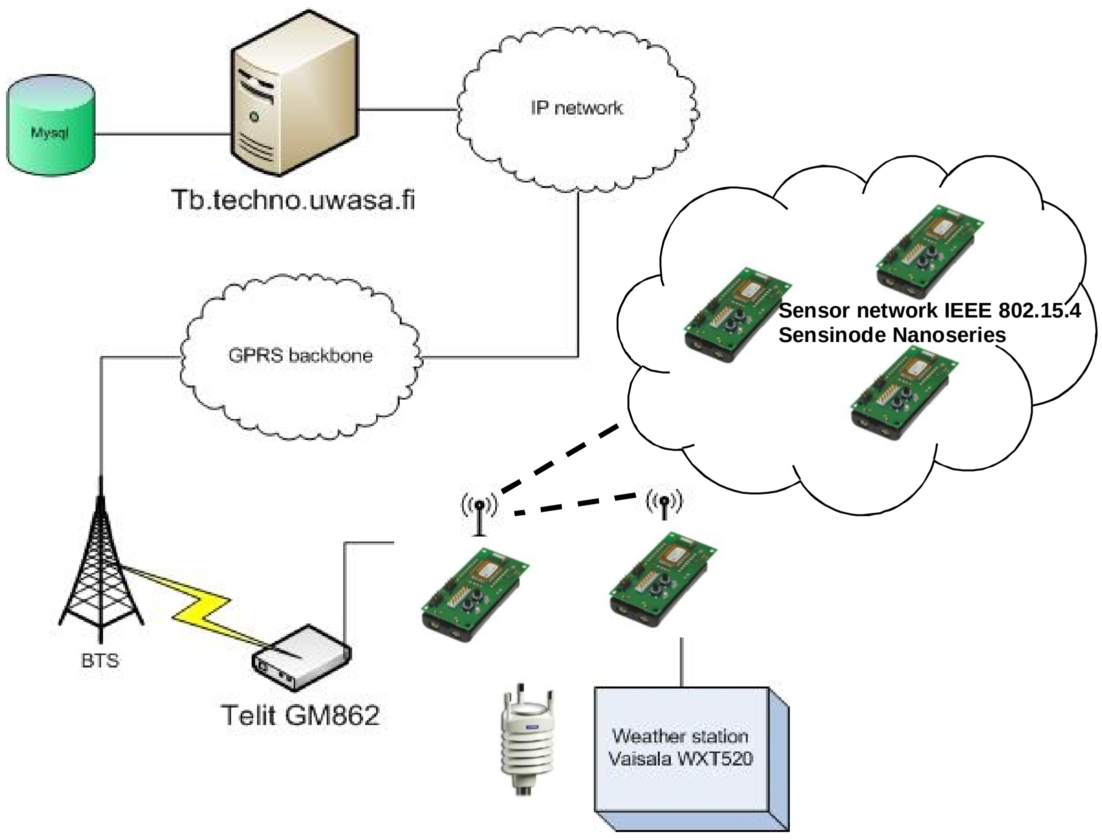


Figure 6.3.2. First version of the system monitoring and control architecture in Söderfjärden [20].

Second version of the monitoring and control architecture has been designed based on the experiences we got when testing the first version and based on the discussions we have had with the research group which is working with the bio-fuels and the diesel generator. In the second version of the system architecture, the UWASA Node is applied. There are also separate channels for monitoring and control data. Monitoring measurements are transmitted by using GPRS and then passed to the university over TCP/IP and stored to the MySQL database. Control commands are transmitted from GPRS to GPRS. The system architecture is illustrated in Figure 6.3.3.

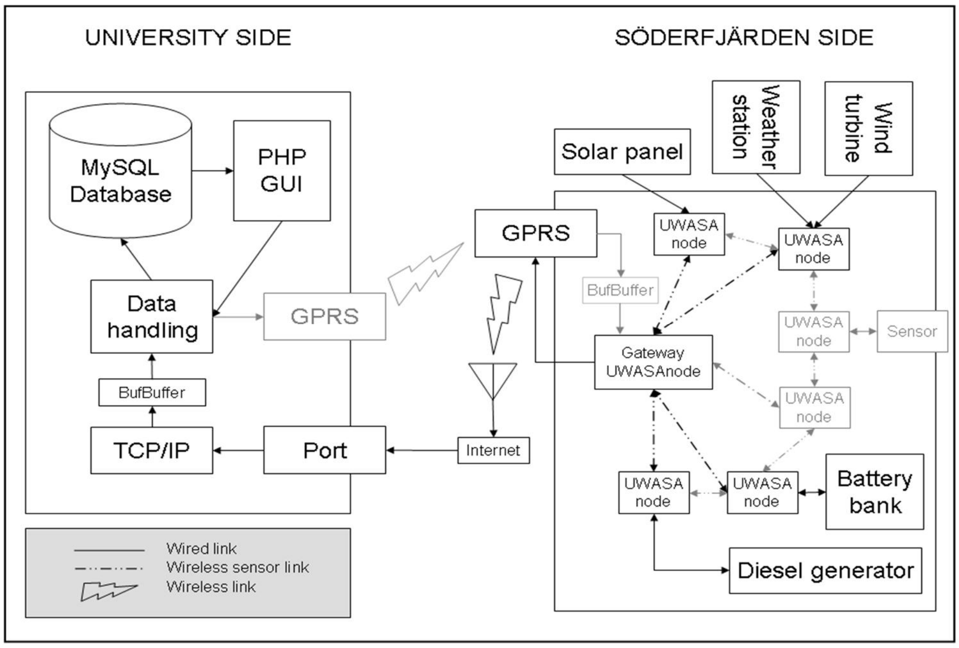
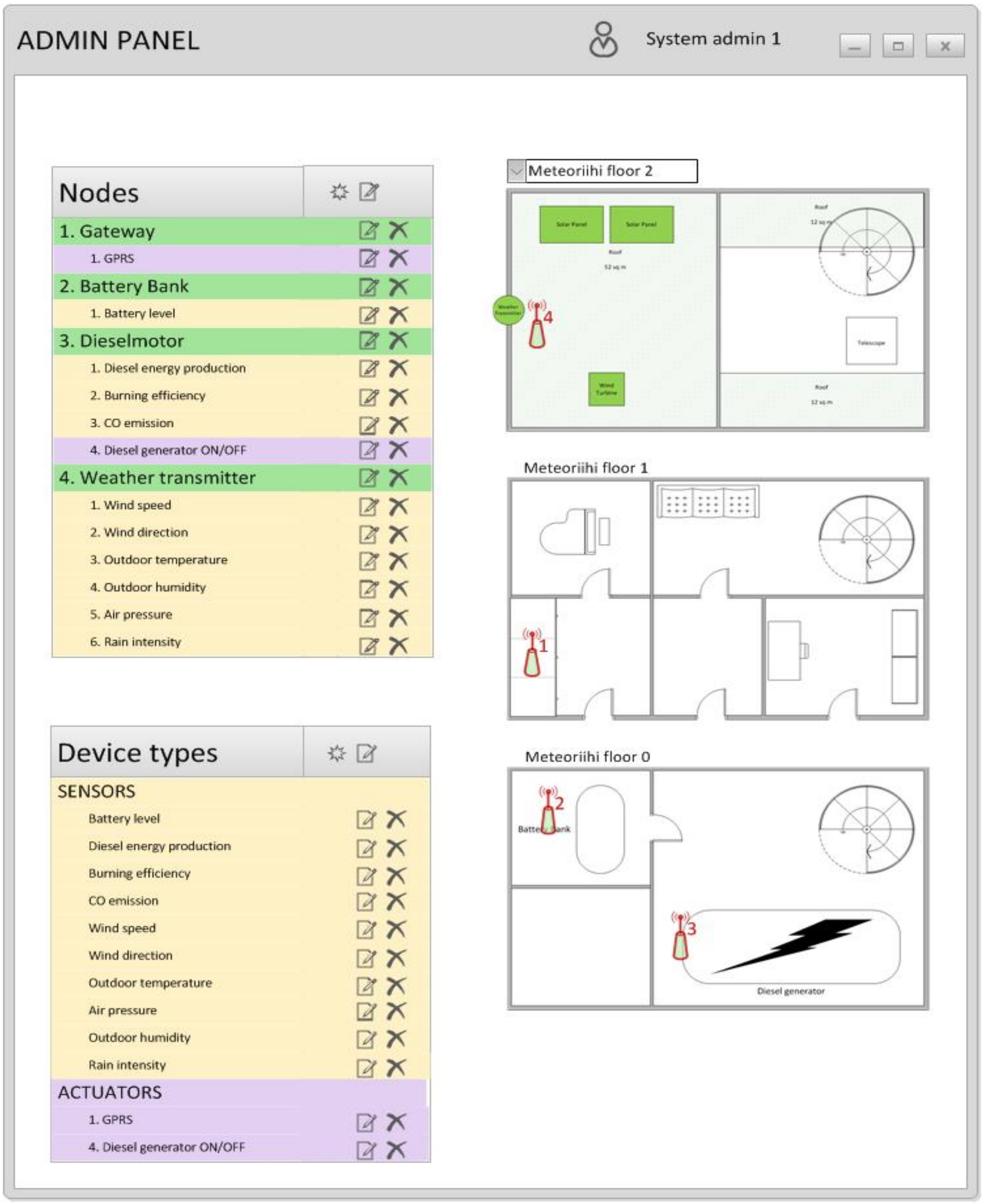


Figure 6.3.3. Second version of the system monitoring and control architecture in Söderfjärden.

More sophisticated application interface (API) is also designed to use the system. The data structures of the long distance links, database and the API are made such that the same solution can be used for several sensor networks monitoring and control applications, not only for Meteoriihi. An example of the developed API is shown in Figure 6.3.4.



ADMIN PANEL

System admin 1

Nodes

1. Gateway
   1. GPRS
2. Battery Bank
   1. Battery level
3. Dieselmotor
   1. Diesel energy production
   2. Burning efficiency
   3. CO emission
   4. Diesel generator ON/OFF
4. Weather transmitter
   1. Wind speed
   2. Wind direction
   3. Outdoor temperature
   4. Outdoor humidity
   5. Air pressure
   6. Rain intensity

Device types

SENSORS

Battery level
Diesel energy production
Burning efficiency
CO emission
Wind speed
Wind direction
Outdoor temperature
Air pressure
Outdoor humidity
Rain intensity

ACTUATORS

1. GPRS
4. Diesel generator ON/OFF

Meteoriihi floor 2

Meteoriihi floor 1

Meteoriihi floor 0

Figure 6.3.4. An example of the API developed for monitoring and control applications.

6.4 Greenhouse Automation

6.4.1 General

Modern greenhouses are so big that having one cabled sensor box in the middle does not anymore provide enough information for reliable control. There are many factors, like differences between north and south end of the greenhouse, differences between areas close to the walls and far away from them and differences between the microclimate layers from top to bottom, which cannot be monitored without several measurement points. More precise information about the conditions inside the greenhouse would also enable more accurate use of the additional carbon dioxide, which is used in greenhouses. It is also needed in so-called closed greenhouses, which are more energy-efficient as the traditional ones and currently under intensive development.

We made our tests at Martens Association's (Martens trädgårdsstiftelse) research greenhouses in Närpiö. The area around Närpiö is the most important greenhouse production area in Finland, and there has been ongoing research cooperation between the University of Vaasa and Martens Association for a while. First WSN test in Närpiö was made in 2008 [21], and based on these experiences the work was continued and extended in GENSEN-project.

6.4.2 Kick-Off Tests in Martens Greenhouse

In the greenhouse tests, twenty nodes equipped with temperature and humidity sensor are deployed in a tomato greenhouse. Due to the height and to the dense foliage of the tomato plants (Figure 6.4.1), this environment makes the propagation of low-power radio signals difficult. Figure 6.4.2 represents the links discovered during the initialization phase and the ones selected to build the tree-like topology.

Figure 6.4.1. The tomato greenhouse.

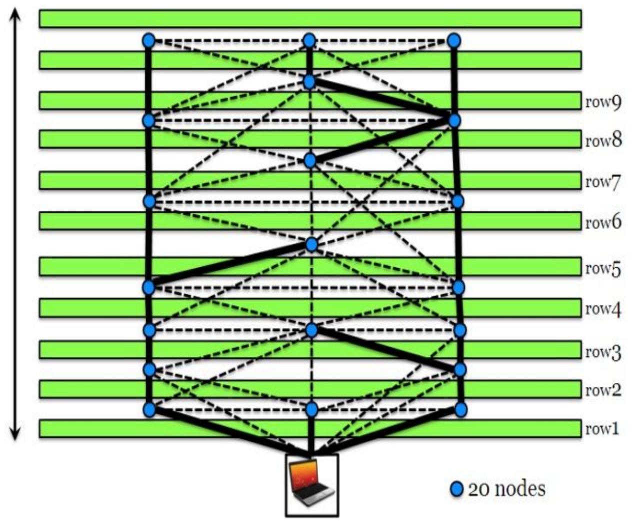


Figure 6.4.2. The three-like topology defined by Kick-Off in the deployment carried out in the greenhouse. The dashed lines represent the links discovered during the initialization phase. The solid lines represent the links selected to build the three-like topology

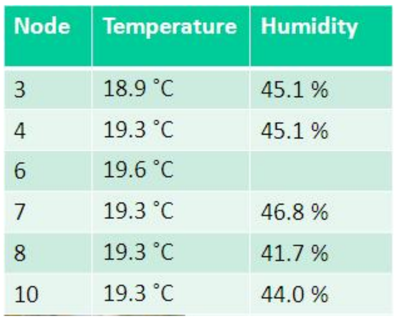
After having fixed the time synchronization bug observed in the first deployment carried out in the industrial hall, each sensor node was required to transmit to the sink node three thousands data packets, one per second. The average PDR of the twenty nodes was 99.78% (99.12% the worse, 100% the best). Each data packet transmitted by a node to its parent in the topology was acknowledged.

In a second test run, each node was required to transmit to the sink node 100 data packets, one every 30 seconds. In this case, the average PDR of the twenty nodes was 99.89% (99.27% the worse, 100% the best).

6.4.3 SurfNet in Greenhouse

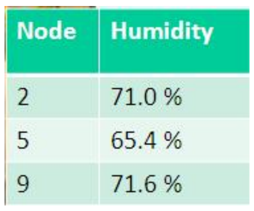
In the greenhouse case the aim of SurfNet tests was to test very low-power sensor functions and wireless range in real environment. SurfNet node had temperature and humidity sensors (Honeywell HIH-5030 and GE sensing MC65F103BA), which were turned on for about 5 ms in 1.0 second periods. The sensor values were uncalibrated. The values were very stable (Table 6.4.1), but there was an error in the humidity sensor calibration; the real humidity in greenhouse was about 70%. Earth surface water content was measured indirectly by air humidity using a closed pipe, which was open inside the earth. This sensor seems to be feasible after calibration (Table 6.4.2).

Table 6.4.1. Measured air condition values.



Node	Temperature	Humidity
3	18.9 ˚C	45.1 %
4	19.3 ˚C	45.1 %
6	19.6 ˚C	
7	19.3 ˚C	46.8 %
8	19.3 ˚C	41.7 %
10	19.3 ˚C	44.0 %

Table 6.4.2. Measured values in indirect earth humidity sensor.



Node	Humidity
2	71.0 %
5	65.4 %
9	71.6 %

The development of the mesh topology software was in its early stage during the time when the greenhouse experiments were made. As a consequence, the ranging tests did not produce as good results as they did later on in the laboratory experiments. In the greenhouse, the reliable wireless communication range seems to be shorter as expected: only 1–5 m, while in open area the range is about 10 m. Tomato foliage and the tomatoes absorbed significant radio signal; specially, as the diameter of a tomato is near the same as the distance of 2.4 GHz antenna.

6.5 Cattle House Automation

6.5.1 General

In modern cattle houses the cows can move freely. During the latest decades the number of cows required to have economically productive herd has increased so much that it is not anymore possible to monitor the cows individually in a same way as it was done during the old times. In this pilot the conditions in the cattle house as well as the locations and activities of the cows were monitored. Both UWASA Nodes and SurfNet Nodes were applied. Because of their small side and low energy consumption, the SurfNet Nodes are suitable to add to the collars the cows wear in any case.

The tests took place at the cattle house of Seinäjoki University of Applied Sciences in Ilmajoki. Earlier a commercial system which applies IEEE 802.11 communication protocol (WLAN) has been tested in same place, now the tests were made by using our IEEE 802.15.4 based architecture.

6.5.2 Kick-Off Tests in Cattle House

In the Cattlehouse test, sixteen nodes equipped with a temperature and humidity sensor were deployed. Compared to the greenhouse, the cattle house is a more open environment: the nodes positioned at the two ends of the deployment area are able to communicate directly. However, the presence of several metallic surfaces, e.g. cages and fences, and of mobile objects, i.e. cows, makes also this environment challenging (see Figure 6.5.1).

In a long test run, each sensor node was required to transmit to the sink node five thousands data packets, one per second. The average PDR of the sixteen nodes was 99.92% (99.43% the worse, 100% the best). Each data packet transmitted by a node to its parent in the topology was acknowledged.

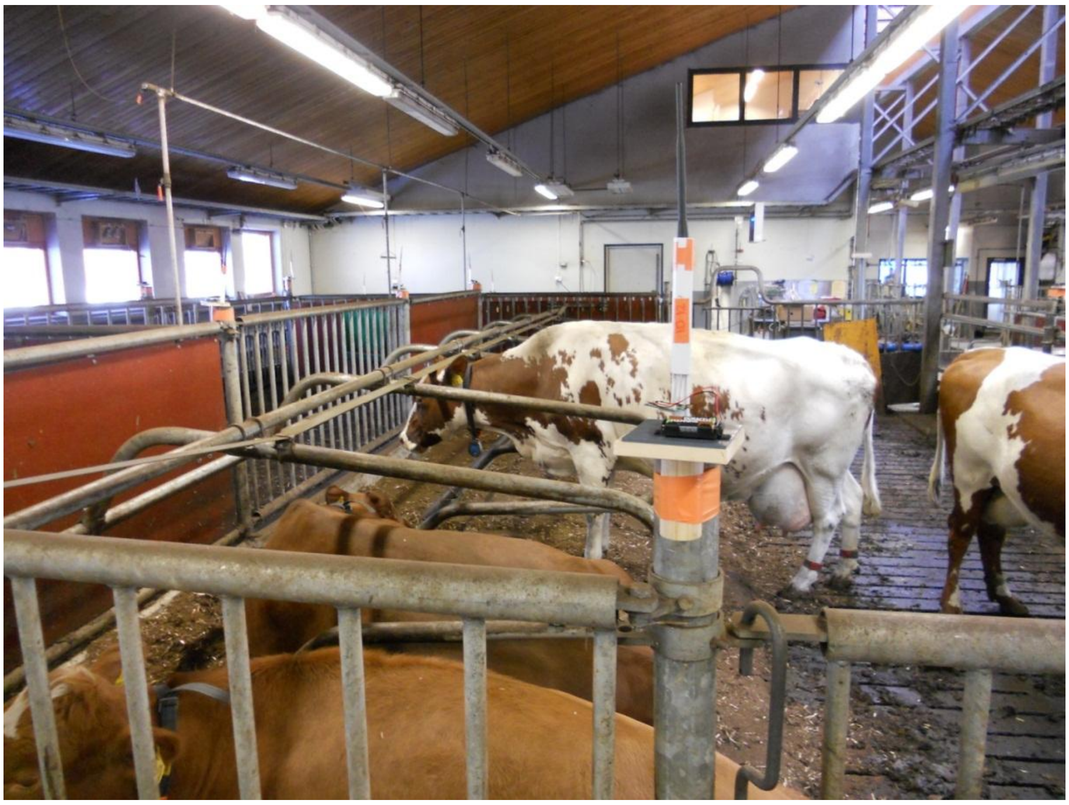


Figure 6.5.1. Test setup at Seinäjoki University of Applied Sciences training and research cattle house in Ilmajoki.

6.5.3 SurfNet in Cattle House

Cattle house provided one realistic environment to test SurfNet operation in mesh network topology, neighbor based positioning and the use of acceleration sensors to monitor the behavior of the cows.

The cattle house was covered by 12 fixed-position anchor nodes in the ceiling. These nodes operated as data routers and they also provided a fixed grid for the positioning of the rest of the nodes (Figure 6.5.2). Air temperature was also measured by the fixed position nodes.

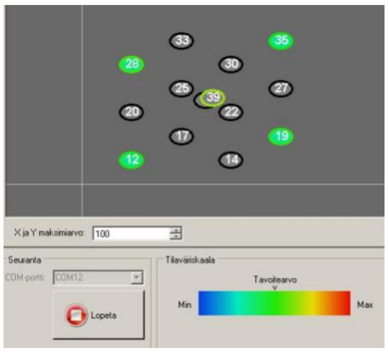
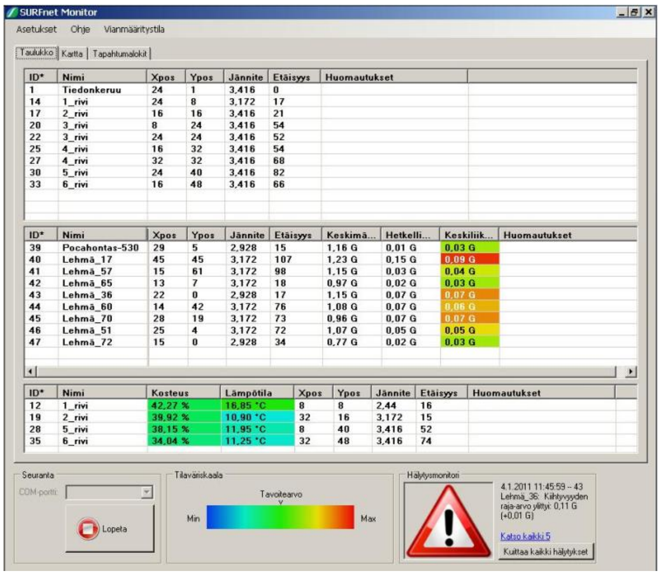


Figure 6.5.2. Fixed location nodes and two positioning nodes.

The sensing and positioning nodes were attached to the collars of the cows. By localizing them we were able to localize the cows, and simultaneously the 3D acceleration was measured. The measured data was collected into PC via bridge node and USB interface. The measured mean acceleration was shown as colors both in graphical display and in table format. Naturally, all measured data was visible also in numeric format. In table format in Figure 6.5.3 first node '1' is a bridge node, next 8 nodes are the fixed position nodes without sensors, next 9 nodes are the sensing nodes attached to the collars of the cows and the last 4 nodes are fixed position nodes with temperature sensor.



ID*	Nimi	Xpos	Ypos	Jännite	Etäisyys	Huomautukset
1	Tiedonkeruu	24	1	3,416	0	
14	1_rivi	24	8	3,172	17	
17	2_rivi	16	16	3,416	21	
20	3_rivi	8	24	3,416	54	
22	3_rivi	24	24	3,416	52	
25	4_rivi	16	32	3,416	54	
27	4_rivi	32	32	3,416	68	
30	5_rivi	24	40	3,416	82	
33	6_rivi	16	48	3,416	66	

ID*	Nimi	Xpos	Ypos	Jännite	Etäisyys	Keskimä...	Hetkelli...	Keskiliik...	Huomautukset
39	Pocahontas-530	29	5	2,928	15	1,16 G	0,01 G	0,03 G	
40	Lehmä_17	45	45	3,172	107	1,23 G	0,15 G	0,09 G	
41	Lehmä_57	15	61	3,172	98	1,15 G	0,03 G	0,04 G	
42	Lehmä_65	13	7	3,172	18	0,97 G	0,02 G	0,03 G	
43	Lehmä_36	22	0	2,928	17	1,15 G	0,07 G	0,07 G	
44	Lehmä_60	14	42	3,172	76	1,08 G	0,07 G	0,06 G	
45	Lehmä_70	28	19	3,172	73	0,96 G	0,07 G	0,07 G	
46	Lehmä_51	25	4	3,172	72	1,07 G	0,05 G	0,05 G	
47	Lehmä_72	15	0	2,928	34	0,77 G	0,02 G	0,03 G	

ID*	Nimi	Kosteus	Lämpötila	Xpos	Ypos	Jännite	Etäisyys	Huomautukset
12	1_rivi	42,27 %	16,85 °C	8	8	2,44	16	
19	2_rivi	39,92 %	10,90 °C	32	16	3,172	15	
28	5_rivi	38,15 %	11,95 °C	8	40	3,416	52	
35	6_rivi	34,04 %	11,25 °C	32	48	3,416	74	

4.1.2011 11:45:59 -- 43
Lehmä_36: Kiihtyvyyden raja-arvo ylittyi: 0,11 G (+0,01 G)

Katso kaikki 5

Kuittaa kaikki hälytykset

Figure 6.5.3. Table format display.

The mesh topology and gossip routing worked quite well. Because of multi-hop routing and slow, power saving synchronization, the measured data was visible in screen after 15 to 60 seconds delay. The iron structures and concrete walls did not absorb the radio signal, but rather vice versa. The neighborhood based positioning was in troubles, because the walls were functioning as plate antennas. This feature is shown in Figure 6.5.4. The object's (a man in this case) position was estimated quite well in the middle of the cattle house, but near the wall the accuracy was weaker. The node reached better farther neighbors and estimated itself to be closer to them what it really was. The difference between estimated and real position is indicated by the red line between the red dot and red circle.

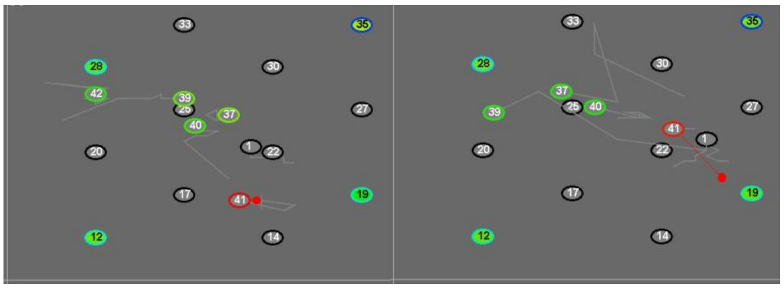
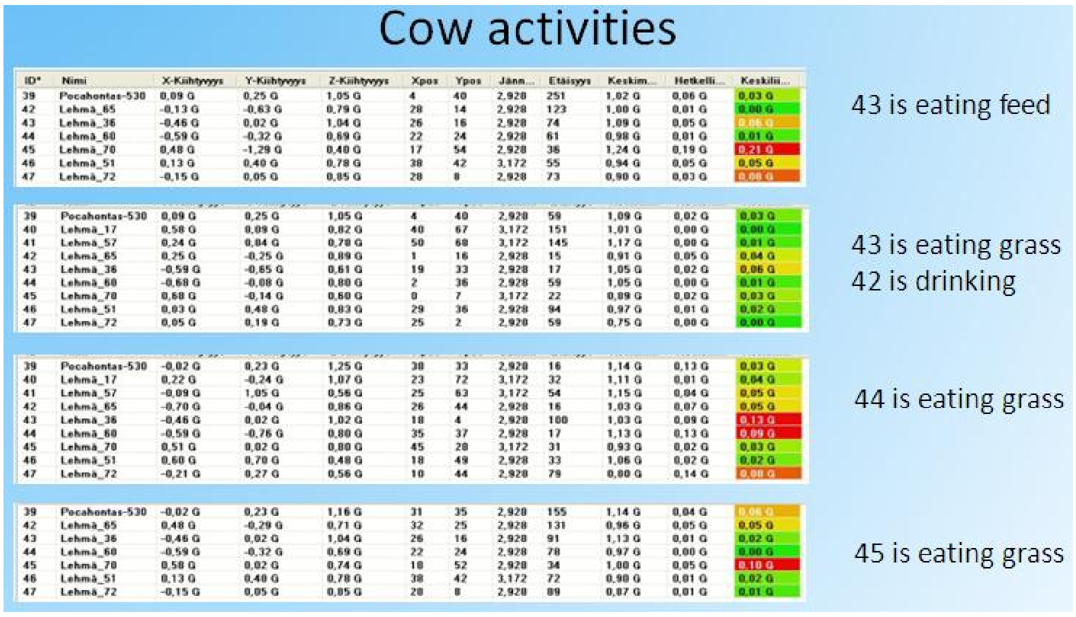


Figure 6.5.4. Neighborhood positioning

The monitoring of the activity of the cows by using acceleration sensors was a little uncertain. The cows were moving very slowly. Moreover, once the cow rested, it chewed and moved its neck, which caused some misleading acceleration to the sensors attached to the collar. However, these results can be remarkably improved by developing further the algorithms to analyze the accelerometer data. Some results we got are shown in Figure 6.5.5.

The system feasibility was demonstrated, but the data analysis part must be developed further.



Cow activities

ID*	Nimi	X-Kiihtyvyys	Y-Kiihtyvyys	Z-Kiihtyvyys	Xpos	Ypos	Jänn...	Etäisyys	Keskim...	Hetkelli...	Keskilii...
39	Pocahontas-530	0,09 G	0,25 G	1,05 G	4	40	2,928	251	1,02 G	0,06 G	0,03 G
42	Lehmä_65	-0,13 G	-0,63 G	0,79 G	28	14	2,928	123	1,00 G	0,01 G	0,00 G
43	Lehmä_36	-0,46 G	0,02 G	1,04 G	26	16	2,928	74	1,09 G	0,05 G	0,06 G
44	Lehmä_60	-0,59 G	-0,32 G	0,69 G	22	24	2,928	61	0,98 G	0,01 G	0,01 G
45	Lehmä_70	0,48 G	-1,29 G	0,40 G	17	54	2,928	36	1,24 G	0,19 G	0,21 G
46	Lehmä_51	0,13 G	0,40 G	0,78 G	38	42	3,172	55	0,94 G	0,05 G	0,05 G
47	Lehmä_72	-0,15 G	0,05 G	0,85 G	28	8	2,928	73	0,90 G	0,03 G	0,08 G

43 is eating feed

39	Pocahontas-530	0,09 G	0,25 G	1,05 G	4	40	2,928	59	1,09 G	0,02 G	0,03 G
40	Lehmä_17	0,58 G	0,09 G	0,82 G	40	67	3,172	151	1,01 G	0,00 G	0,00 G
41	Lehmä_57	0,24 G	0,84 G	0,78 G	50	68	3,172	145	1,17 G	0,00 G	0,01 G
42	Lehmä_65	0,25 G	-0,25 G	0,89 G	1	16	2,928	15	0,91 G	0,05 G	0,04 G
43	Lehmä_36	-0,59 G	-0,65 G	0,61 G	19	33	2,928	17	1,05 G	0,02 G	0,06 G
44	Lehmä_60	-0,68 G	-0,08 G	0,80 G	2	36	2,928	59	1,05 G	0,00 G	0,01 G
45	Lehmä_70	0,68 G	-0,14 G	0,60 G	0	7	3,172	22	0,89 G	0,02 G	0,03 G
46	Lehmä_51	0,03 G	0,48 G	0,83 G	29	36	2,928	94	0,97 G	0,01 G	0,02 G
47	Lehmä_72	0,05 G	0,19 G	0,73 G	25	2	2,928	59	0,75 G	0,00 G	0,00 G

43 is eating grass
42 is drinking

39	Pocahontas-530	-0,02 G	0,23 G	1,25 G	30	33	2,928	16	1,14 G	0,13 G	0,03 G
40	Lehmä_17	0,22 G	-0,24 G	1,07 G	23	72	3,172	32	1,11 G	0,01 G	0,04 G
41	Lehmä_57	-0,09 G	1,05 G	0,56 G	25	63	3,172	54	1,15 G	0,04 G	0,05 G
42	Lehmä_65	-0,70 G	-0,04 G	0,86 G	26	44	2,928	16	1,03 G	0,07 G	0,05 G
43	Lehmä_36	-0,46 G	0,02 G	1,02 G	18	4	2,928	100	1,03 G	0,09 G	0,13 G
44	Lehmä_60	-0,59 G	-0,76 G	0,80 G	35	37	2,928	17	1,13 G	0,13 G	0,09 G
45	Lehmä_70	0,51 G	0,02 G	0,80 G	45	28	3,172	31	0,93 G	0,02 G	0,03 G
46	Lehmä_51	0,60 G	0,70 G	0,48 G	18	49	2,928	33	1,06 G	0,02 G	0,02 G
47	Lehmä_72	-0,21 G	0,27 G	0,56 G	10	44	2,928	79	0,80 G	0,14 G	0,08 G

44 is eating grass

39	Pocahontas-530	-0,02 G	0,23 G	1,16 G	31	35	2,928	155	1,14 G	0,04 G	0,06 G
42	Lehmä_65	0,48 G	-0,29 G	0,71 G	32	25	2,928	131	0,96 G	0,05 G	0,05 G
43	Lehmä_36	-0,46 G	0,02 G	1,04 G	26	16	2,928	91	1,13 G	0,01 G	0,02 G
44	Lehmä_60	-0,59 G	-0,32 G	0,69 G	22	24	2,928	78	0,97 G	0,00 G	0,00 G
45	Lehmä_70	0,58 G	0,02 G	0,74 G	18	52	2,928	34	1,00 G	0,05 G	0,10 G
46	Lehmä_51	0,13 G	0,40 G	0,78 G	38	42	3,172	72	0,90 G	0,01 G	0,02 G
47	Lehmä_72	-0,15 G	0,05 G	0,85 G	28	8	2,928	89	0,87 G	0,01 G	0,01 G

45 is eating grass

Figure 6.5.5. Measured cow activities.

7 CONCLUSIONS

In this project we considered IEEE 802.15.4 communication protocol and 2.4 GHz license-free band. We evaluated the current state of the art of wireless sensor networks in wireless automation, developed a generic architecture and evaluated its performance through five test cases. The market situation and user requirements were evaluated by a literature overview and by interviewing key persons from the leading companies and research institutes operating in the area. Based on the results and on the ideas of the research consortium, steps from platform design to commercialization are presented.

A generic software and hardware architecture was designed and its performance was evaluated trough five different test cases. Developed architecture consists of protocol software called A-Stack and two sensor platforms: UWASA Node and SurfNet. These platforms can be used independently or jointly. The outcomes of GENSEN-project have been used in the preparation of project Reliable and Real Time Wireless Automation (RIWA), which is funded from Tekes Ubicom program and started in June 2011.

In addition to wireless automation, the architecture developed in GENSEN will also be used in tactical communication systems research by the same research partners. Other applications where the results of GENSEN are currently utilized for further research are some healthcare applications by Seinäjoki University of Applied Sciences and some machinery applications by the University of Vaasa.

REFERENCES

[1] HART Communication Foundation, http://www.hartcomm.org/
[2] The International Society of Automation, http://www.isa.org/
[3] Dust Networks Inc., http://www.dustnetworks.com/
[4] Saaranen, J. and Keskinen, S., *Product and Process Platform – a New Concept for Management*, 7th International Conference on Productivity & Quality Research (ICPQR'98), Miami, USA, 1998.
[5] R. Virrankoski and S. Keskinen, *GENSEN: A Novel Combination of Product, Application and Technology Platform Development in the Context of Wireless Automation*, in the proceedings of 14th International Conference on Productivity & Quality Research (ICPQR 2009), October 19-23, 2009, Alexandria, Egypt.
[6] Verho, A. and Salminen, V., *Tuotekehityksen nopeuttaminen*, Konepajamies 47:3, ss. 28-30, 1994.
[7] Wheelwright, S. C. and Clark, K. B., *Creating Project Plans to Focus Product Development*, Harvard Business Review, 1992.
[8] Moore, G. A., Johanson, P. and Kippola, T., *The Gorilla Game: an Investor's Guide to Picking Winners in High Technology*, 1999.
[9] The FreeRTOS Project, http://www.freertos.org/
[10] Cosar, E.I., Mahmood, A. and Björkbom, M., A-*Stack: A Real-Time Protocol Stack for IEEE 802.15.4 Radios*, to appear in IEEE SenseApp, 2011.
[11] Cosar, E.I. and Mahmood, A., *A-Stack Documentation*, Aalto University Laboratory Report, 2011.
[12] Poor, R. D., *Reliable wireless networks for industrial applications,* In B. Fette, A. Bensky, P. Chandra, & D. Dobkin, RF & Wireless Technologies (pp. 423-434).
[13] Yigitler, H., *The UWASA Node Reference Manual*, University of Vaasa, Department of Computer Science, Communications and Systems Engineering Group, 2010.
[14] Yigitler, H., Virrankoski, R. and Elmusrati, M. S., *Stackable Wireless Sensor and Actuator Network Platform for Wireless Automation: The UWASA Node*, Aalto University Workshop on Wireless Sensor Systems (WoWSS), November 19th, 2010, Espoo, Finland.
[15] SeAMK Embedded Systems Resouce Page, http://lompsa.seamk.fi/sulautetut/
[16] H. Palomäki, *Wireless Network in Ambient Intelligence*, Licentiate of Science Thesis, Tampere, Finland: Tampere University of Technology, 2008.
[17] M. Huhta, *Radiopiirin ohjemointiympäristön kehitys ja langattoman verkon tahdistus*, Bachelor's Thesis, Seinäjoki University of Applied Sciences, School of Technology, 2009.
[18] H. Palomäki and M. Huhta, *Low Power Synchronization in Wireless Network*, 2nd Workshop on Wireless Communication and Applications (WoWCA2010), Vaasa, Finland, 2010.

[19] PROPSim, http://www.elektrobit.com/what_we_deliver/wireless/offering/propsim_f8

[20] Wik, N., *Weather LAN – A Local Area Network for Monitoring and Control*, Master's Thesis, University of Vaasa, Department of Computer Science, Communications and Systems Engineering Group, 2009.

[21] Ahonen, T., Virrankoski, R. and Elmusrati, M. S., *Greenhouse Monitoring with Wireless Sensor Network*, in the proceedings of 2008 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, October 12-15, 2008, Beijing, China.

[22] Bocca, M. and Koivo, H., *Kick-Off: an Initialization Procedure for Convergecast in Wireless Sensor Networks*, Submitted to 7th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), December 6-9, 2011, Adelaide, Australia.

APPENDIX

Other publications produced in the context of GENSEN-project but not mentioned in the reference list of this report:

Björkbom, M., Eriksson, L. and Silvo, J., *Technologies and Methodologies Enabling Reliable Real-Time Wireless Automation*, New Trends in Technologies: Control, Management, Computational Intelligence and Network Systems, Meng Joo Er (Ed.), ISBN: 978-953-307-213-5, Sciyo 2010.

Cosar, E.I., Mahmood, A., Björkbom, M., *A-Stack: A Real-Time Protocol Stack for IEEE 802.15.4 Radios*, in IEEE SenseApp 2011, Bonn, Germany, October 2011 (short paper).

Cosar, E.I. and Mahmood, A., *A-Stack Documentation*, Aalto University Laboratory Report, 2011.

Mahmood, A. and Jäntti, R., *A Decision Theoretic Approach for Channel Ranking in Crowded Unlicensed Bands*, Wireless Networks, Vol. 17, No. 3, May 2011, pp. 907-919.

Bocca, M., Kaltiokallio, O. and Silvo, J., *Real-Time and Reliable Wireless Sensor Networks for Smart Wireless Automation Applications*, Suomen Automaatioseura seminar AutomaatioXIX 2011, Helsinki, March 2011.

JValarezo, J., Silvo, J. and Jäntti, R., *Enhancing RSSI-Based Ranging for Localization of IEEE 802.15.4 Devices in Indoor Situation*, In Proc. WoWCA 2010, Vaasa, Finland, 2010.

Björkbom, M., Jäntti, R. and Koivo, H., *Real-time wireless sensor systems for monitoring and control*, in Automaatio XIX SEMINAARI 15.-16.3.2011, Arto Visala (toim.), Helsinki: Suomen Automaatioseura ry, 2011, 6 pp.

Eriksson, L., Silvo, J., Björkbom, M., Nieminen, J. and Jäntti, R. (in Finnish), *Kohti reaaliaikaista ja luotettavaa langatonta automaatiota*, Automaatioväylä, Vol. 3, s. 24-27, 2011.

Doctoral Dissertations:

Maurizio Bocca, *Application-Driven Data Processing in Wireless Sensor Networks*, Doctoral Dissertation, Aalto University School of Electrical Engineering, Espoo, Finland, November 2011.

Master’s Thesis:

Joni Silvo, Reliable communication protocol for real-time wireless automation, Master’s Thesis, Aalto University, School of Science and Technology, 2011.

Jose Valarezo, *Enabling Wireless Sensor Localization in Dynamic Indoor Environments*, Master's Thesis, Aalto University School of Science and Technology, 2011.