

# Designing Wireless Sensor Technology

Ventocom – Menno Mennenga

Requirement + Test Engineering

Arndtstrasse 9, D – 01099 Dresden

Email: menno.mennenga@ventocom.com

Phone: +49 173 565 8134

## 1 Abstract

Technologies for Wireless Sensor Networks (WSN) have generated considerable interest among Embedded Developers. Freedom from wires promises easy deployment, reduced maintenance, increased flexibility, additional comfort and reduced cost in gathering up-to-the-second information. These advantages motivated engineers to develop a large number of exciting products over the last years, based on open-standard technologies such as IEEE 802.15.4, ZigBee, WirelessHART, or 6lowPAN.

WSN are, by definition, distributed, parallel networks of nodes communicating over wireless links. Engineers dealing with such systems are aware that distributed development is a complex task. This paper intends to unveil some of the complexity by giving insight into the engineering of WSN applications based on previous, real-world experience.

This paper uses the product life cycle model to frame the challenges and potential solutions in WSN development. Three life cycle phases (conception, design and service) are considered. Specifically, categories for requirements collection during the conception phase are presented. Important points of the realization phase such as the WSN system architecture, implementation and test are discussed. Finally, this paper includes requirements and implementation options of the service phase (installation, operation, and maintenance).

## 2 Introduction

The Ventocom team has been helping to bring WSN technology into reality since 2004. Initially, the focus was on implementing IEEE802.15.4 ICs. At a later stage, activities included 15.4 and Zigbee application development. Over time, it was realized that even though WSN applications were quite varied, the development process typically followed a certain pattern. Without claiming to be completely exhaustive, this paper intends to summarize the most important steps of this process. It is hoped that it will be valuable to R&D managers and engineers alike in the continuing adoption of WSN technology.

The paper is organized as follows. Section 3 describes a number of assumptions about the “typical” WSN project and it introduces the phases of the product life cycle (PLC) model. The rest of the paper is organized along three of the PLC phases: Section 4 summarizes the collection of requirements in the conception phase, section 5 guides the reader through the implementation phase (including architecture decisions, system implementation and test), and section 6 is dedicated to the service phase (system installation, operation and maintenance).

### 3 Assumptions

For the purpose of this paper, it is assumed that a WSN project starts with the decision to develop a product that will incorporate WSN technology. It is also assumed that an open-standard WSN protocol stack will be used (candidates being IEEE802.15.4, ZigBee, WirelessHART, or 6lowPAN), but that a firm choice has yet to be made. The stack development will not be in the focus of the development effort. Furthermore, it is assumed that the development team plans to use an IEEE802.15.4-compliant transceiver.

Another assumption is that the project will follow the PLC phases shown in Figure 1. Clearly, a real-world project will not always rigidly follow this pattern, but for the purpose of this paper it was found to be useful in structuring the description of the development process. The remaining sections are organized along these phases; only the realization phase has been omitted.

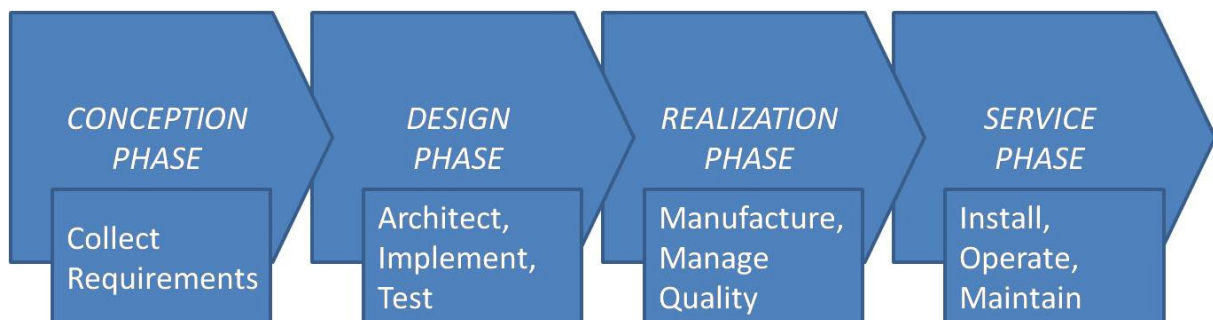


Figure 1: Phases of the Product Life Cycle Model

### 4 Conception Phase

Development activities begin with the conception phase, where engineers and their colleagues from other departments (such as marketing and sales) jointly collect product requirements.

Aspects to be considered by that team include (1) system performance, (2) integration with other systems, (3) system behavior in the service phase (where the product will be installed, operated and maintained), (4) regulatory requirements, and (5) safety-related requirements. Each aspect will be considered in detail in the subsequent sections.

#### 4.1 Performance Requirements

Performance requirements for WSN include target cost (both development and part cost), number of parts annually produced, number of nodes in the network, size of coverage area, battery life time (if nodes are not mains powered), throughput, response time, duty cycle, and security requirements (encryption and/or authentication).

The relationship between these performance requirements and the corresponding system architecture decisions will be described in section 5.1.

## **4.2 Integration Requirements**

WSN need to be integrated with sensors, actors, control panels, displays or gateways. Integration requirements are documented in the form of mechanical and electrical characteristics of their interfaces, as well as communication protocol requirements.

In addition, for gateway interfaces, protocol conversion rules need to be defined – for example when a ZigBee network shall be connected to an IP network, a field bus, or a wide-area network such as GSM/GPRS or WiMAX. Less common requirements include the integration with digital signal processing (software) modules such as codecs for image or voice transmission.

## **4.3 Use Case Requirements**

It is advantageous to consider typical activities in the service phase to derive use cases for WSN products, such as installation, network formation, security or authentication, key distribution, regular operation, graceful recovery from temporary interference, detection and removal of a malfunctioning device, over-the-air firmware upgrades, and over-the-air configuration.

Collecting the use cases is usually the most complex task in the conception phase. A detailed look at these requirements and potential implementations is taken in section 6.

## **4.4 Regulatory Requirements**

Embedded designers are familiar with regulatory requirements imposed by the prerequisites for CE marking such as the EMC requirements of the low-voltage directive or the machinery directive and the relevant standards. Less obvious regulatory requirements for WSN are radio power emission and interference limits set by ETSI and the FCC. It should also be noted that for wireless systems, CE marking imposes special EMC requirements.

Decisions need to be taken how to ensure compliance with regulatory requirements (testing of prototype vs. production-line testing, in-house testing vs. cooperation with a test specialist). This point should be thoroughly discussed especially when sourcing radio modules from an external supplier.

## **4.5 Safety-Related Requirements**

A point which originally arises from regulatory requirements in relationship with CE marking is product risk analysis. For this activity, manufacturers need to consider the reliability of the wireless links in WSN. In spite of claims to the contrary, the performance of wireless links may suffer in the presence of interference, and recovery mechanisms such as channel hopping may not always overcome the performance drop. Risk analysis exercises should therefore consider situations where a wireless link is unavailable, in analogy to cable defects. Corresponding failure scenarios must be considered in risk analysis, and detection and recovery mechanisms should be specified and implemented.

# **5 Design Phase**

Subsequent to conception, this section describes the activities of the design phase. They include architecture design, the implementation of electronic components, software and mechanical design, as well as test activities.

## **5.1 Architecture Decisions**

The most important architecture decisions in WSN development include

- The choice of a protocol software alternative (IEEE802.15.4, ZigBee, WirelessHART, or a flavor of the 6lowPAN protocol)
- The choice of a single processor or separate processors for stack and application
- If applicable, considerations for battery-powered operation
- If applicable, a decision on the method of achieving long range

Another important decision is the selection of the system platform supplier (chip set or radio module) with considerations to price, protocol stack choices, frequency band (2.4GHz and sub-1-GHz), power consumption and other factors.

The decisions summarized in the list above can be derived from the performance requirements described in section 4.1. Sections 5.1.1 through 5.1.5 describe that process in detail.

### 5.1.1 Star vs. Mesh Topology

The number of nodes in a network and the size of the coverage area drive the choice of network topology: star network or mesh network. A star network can be realized with an IEEE802.15.4 protocol stack. Mesh networks based on IEEE802.15.4 (ZigBee, WirelessHART, 6lowPAN) have, roughly speaking, the additional capability of transmitting messages over alternative paths (routing). A star and a mesh topology are contrasted in Figure 2.

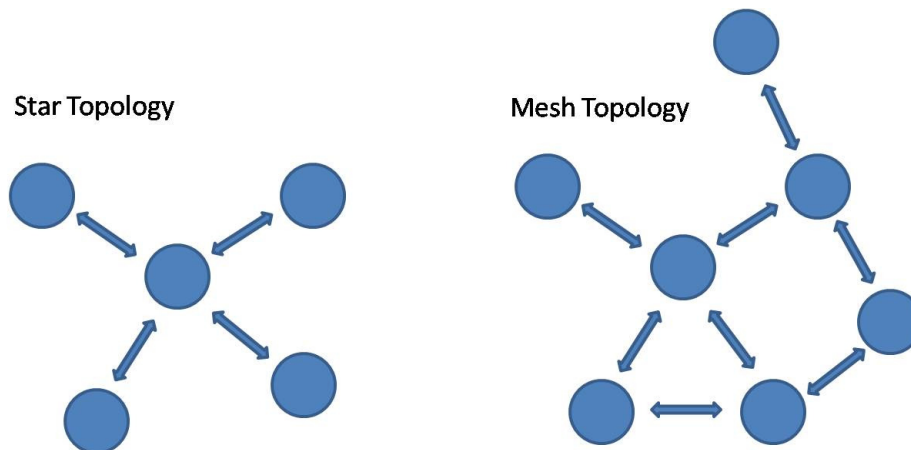


Figure 2: Typical Network Topologies

A star topology is sufficient if two conditions are satisfied: only a small number of nodes (in the magnitude of ten) communicate with each other, and the coverage area does not extend beyond the radio transmission range of say 30 meters in a building. In this case, the technology choice will be IEEE802.15.4. Note that the coverage area can be increased by increasing the transmit power through power amplifiers (also refer to section 5.1.5). Also note that even though the number per network should be small, many such networks may coexist, resulting in a number of nodes per single location much larger than ten. Consider, for example, a battery charge application for fork lift trucks. Each network has three nodes (charger, battery controller, fork lift truck) coexisting in a battery charge room with up to 200 chargers, resulting in 600 nodes in a single location.

For networks with a larger number of nodes, or networks that cover an area beyond the range of a radio link, a mesh topology is the appropriate choice. Mesh networks have the capability of automatically routing data over several hops and can therefore cover a large area containing up to several hundred nodes.

### 5.1.2 Choice of Mesh Technology

Has a designer decided to use an open-standard mesh topology, a choice needs to be made between ZigBee, WirelessHART and 6lowPAN. Each standard offers features that are specifically tailored to the needs of particular industries. ZigBee, for example, comes with stack profiles for home automation and smart energy that will simplify the design of home automation or metering products. WirelessHART offers a process variables interface similar to that of field busses or CANopen, so it may be the natural choice for industrial or automotive applications. Finally, 6lowPAN protocols are very compact and offer a natural integration path with IP-based gateways.

### 5.1.3 Integrated Controller vs. Network/Application Controllers

The software of a WSN node can be partitioned into a communication component (the protocol stack) and an application component (for example for processing of sensor data). The components can be integrated onto a single microcontroller (MCU), or they can be separately implemented on at least two MCUs.

For newly designed products, a single controller for stack and application may be chosen, especially if cost targets are aggressive. Memory and processing power need to be sufficient to support both software components.

If WSN capability is added to an existing product, using separate processors for the WSN stack and the application will be the obvious choice (unless the product is extremely cost-sensitive), especially when a radio module (with transceiver and “network” controller) is used. Figure 3 shows the architecture of such a system. If network controller and application come from different manufacturers, it has the additional benefit of allowing the designer to stay with her choice of application controller and its development tools.

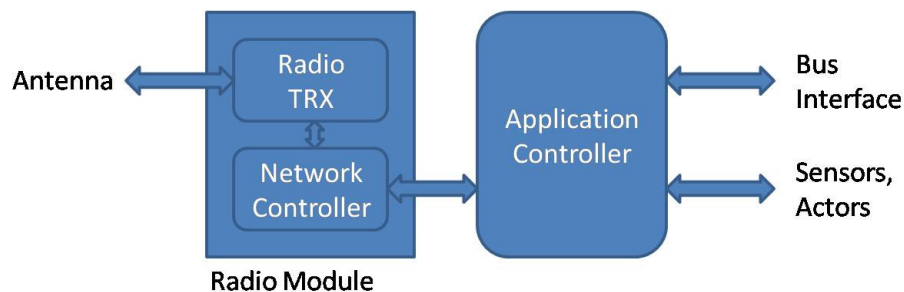


Figure 3: "Split" Architecture with Network Controller and Application Controller

### 5.1.4 Considerations for Battery-Powered Systems

Battery-powered nodes in WSN achieve multi-year battery life by implementing a duty cycle alternating between short active periods for measurement and data transmission, and long inactive periods where the power consumption is extremely low. Battery life times can be calculated using the duty cycle period, the power consumption during the active and sleep periods, and the battery charge.

An important consideration in battery-powered systems is to avoid potential buffer starvation scenarios. Consider a WSN with sensor nodes (data sources) and actor nodes (data sinks). Clearly, an actor can only receive data from its router if it does not “sleep” to conserve battery power. But as long as the actor has not woken up, its router node needs to buffer messages addressed to the actor. If the buffer size of the router device is too small, the arrival rate of messages too high, or time-out periods too short, messages will be lost. In such cases, duty cycle periods, buffer sizes, latency time, transmission-time-out periods and battery life need to be carefully managed to achieve an optimum solution.

Other considerations for battery-powered systems include the capability of activating a node only prior to installation using an on/off switch, for example a hall switch, or other alternatives to a regular on/off switch. Also, battery status messages should be generated and evaluated to enable preventive maintenance.

### 5.1.5 Achieving Long Range

In some cases, WSN need to cover large distances with relatively few nodes, that is, distances larger than the ordinary radio transmission range. Designers may then have to extend the range of the nodes by adding power amplifiers (for higher transmission power) and low-noise amplifiers (for higher receiver sensitivity). Alternatively, if the nodes use a mesh protocol, additional nodes can be installed to forward data between nodes that otherwise would be out of range. Both approaches can also be combined.

## 5.2 Implementation

The implementation commences after fundamental architecture decisions have been made. This section highlights two points with regards to the software application: the support of parallel transactions, and the implementation of error recovery features.

Consider a gateway interface where information enters and leaves the WSN. The upper graph in Figure 4 (depicting the interface between the protocol and the application software modules) shows a data request entering the gateway from the outside. After a (variable) latency period, the network responds with the requested information. The lower graphic shows data requests for node B entering the network right after a data request for node A has been submitted – while the gateway is still waiting for the response of node A. The application software needs to be capable to support the handling of such “parallel” transactions. Note that the responses of the protocol may also be received out of order, for example in the sequence request A, request B, response B, and response A. The application software should be able to handle such sequence inversions as well.

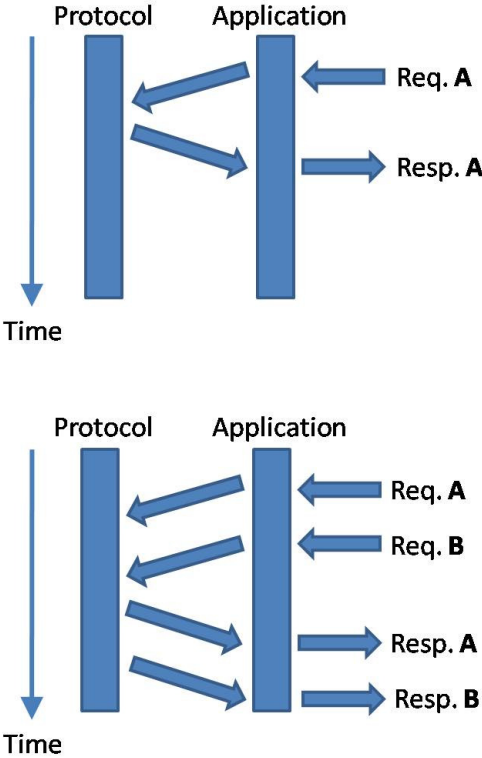


Figure 4: Regular and Parallel Request Processing

Another consideration is the recovery process initiated after network connections have been temporarily unavailable due to temporary interference or a power black-out. Such procedures involve the transmission of special data frames requesting the coordinator to allow a particular node to rejoin the network. If many nodes attempt to rejoin at the same time, communication may fail due to “channel crowdedness”. In that case, the attempts have to be several times repeated, leading to long recovery periods and to unnecessary battery drain. Therefore, the application software should space out in time the attempts to retransmit data or to rejoin the network.

### **5.3 Testing WSN applications**

This section describes the different stages of the test process and highlights WSN-specific challenges arising especially from the distributed nature of WSN applications. Therefore, testing methodologies partially differ from embedded software design for a single microcontroller.

#### **5.3.1 Unit Test**

In section 3, it was assumed that the development engineer will focus on the application software, and rely on off-the-shelf solutions for the protocol software. Consequently, unit testing will concentrate on the application components. Typical test objects are the software modules for sensor data acquisition, data processing, control logic, and output generation.

A unit test framework for the chosen programming language will typically be used. Test automation may be possible if the framework offers automation capabilities. Otherwise, a simple batch program may be developed to run the unit tests automatically.

#### **5.3.2 Integration Test**

The integration test checks if the software components of a single network node correctly work together. The test objects are the interfaces between the application modules, the WSN stack and, if applicable, the gateway interfaces (such as GSM, WiMAX, field buses).

The test system needs to stimulate the test objects by modeling the surrounding network. The test implementation is typically based on an MCU simulator delivered by the MCU manufacturer. Running the integration test on a real-time target like an evaluation board will be faster, but the observation opportunities will be limited.

#### **5.3.3 System Integration Test**

The system integration test extends the test scope beyond a single node. The test object of the system integration test is a network of several nodes. A test system requires the concurrent operation of several network nodes in parallel. Simulators or real-time targets can be used to implement the test system.

Simulators, for example based on SystemC and ns-2, allow the tester to observe all interface and system states (white box test). Running the system integration test on real microcontrollers may be less convenient as usually only the interfaces of the test system can be observed (for example, the serial ports of the network nodes and the radio channel – black box test), on the other hand such a system will run the tests much faster than a simulator can do.

A system integration test setup (regardless if based on real-time targets or on a simulator) can be easily coupled with an automated test system to stimulate the network, to observe the responses of the network nodes, and to automatically form a verdict for each test case.

#### **5.3.4 Performance Test**

Another important test task is the performance test. This test measures the link throughput over the network load (arrival rate of transmission requests). The rate of transmission requests is gradually increased and the rate of successful transmissions is observed.

The implementation effort of a performance test is limited if an automated test system as described in section 5.3.3 is already available. The tester will be rewarded with a large gain of information about the stability of the system and with considerable confidence in the implementation once the performance test has successfully passed.

#### **5.3.5 Field Test**

No lesson in testing is older than “the field test cannot start too early”. Testers should start field trials even if lab tests have not been concluded. No test is better suited to uncover “white spots” in the specification than potential end users working with the prototype. The specification will be quickly improved by uncovering “white spots”, and the tester will be able to build better test systems, eventually reaching a higher confidence in the implementation.

#### **5.3.6 Certification (Pre-) Testing**

Certification testing is required if for example products shall be ZigBee-certified. Compliance tests by an accredited test house will cost several thousand Euros, and it is unfortunate if this test has to be repeatedly run. A tester should therefore become familiar with the testing requirement in such cases and run tests emulating the testing at the test house. Another consideration may be to work with a supplier experienced in testing devices prior to certification tests.

### **6 Service Phase**

Once products are installed, used, and maintained in the service phase, development activities have of course been concluded. But, as mentioned in section 4.3, this part of the product life cycle is discussed to give the reader some insight into possible solutions for challenges that arise from this phase, in order to assist in the requirements collection phase.

#### **6.1 Network Configuration**

Many applications require that only a certain set of nodes may join the network. For example, a wireless ordering system for restaurants needs to ensure that only the nodes in restaurant A may participate in network A, but not the nodes of the adjacent restaurant B. Network access control in ZigBee networks is regulated through the coordinator node’s access table.

The access table can be configured at the manufacturing facility or in the field. Configuration at the manufacturing facility requires prior knowledge of the network size and types of network nodes. Therefore, this approach will only be feasible for standardized products.

Setting the network configuration in the field is more flexible in that respect but it requires a process (and sometimes a tool for the installation technician) to identify unique IDs of the network nodes and to program the access table. Potential solutions may include bringing coordinator and joining node into a special state (by pressing buttons on each) causing the ID of the new node to be transmitted to the coordinator and stored there. The same process can be used to exchange other important configuration information such as encryption and authentication keys that will be used to secure the network traffic against deciphering and tampering. Ideally, installation tools would also contain a means of documenting the serial numbers and locations of network nodes to simplify the identification of malfunctioning devices.

## 6.2 Network Forming

Network forming and route establishment are automatic processes in mesh networks (such as ZigBee) that do not require user involvement. If temporary interference causes interruptions of the network traffic, the frequency channel will be changed and routes will be reestablished.

In star-topology networks, network forming is performed by executing a standardized association procedure between the child nodes and the coordinator. In that case, if the application is based on IEEE802.15.4, it is responsible to implement network recovery and channel switching, which in mesh networks are implemented in the protocol software.

## 6.3 Maintenance

Cost-effective maintenance of a WSN often requires the capability of remotely updating the node software, the monitoring of the network for broken devices or low batteries, and the easy identification of a broken device by a maintenance technician. The following sections discuss the implementation aspects of these features.

### 6.3.1 Software Update

Usually, the software of a network node is stored in the internal flash memory of its microcontroller. Software updates require the replacement of the software with new code. Updating the software through a wireless link can be achieved by one of the following methods:

- Downloading the complete code into external flash memory, then replacing the contents of the internal flash by the contents of external flash (requires a boot loader)
- Downloading the new code directly into internal flash (requires a loader that is capable of network transmission)

While the latter method does not require external flash memory, it is only recommended for situations where it is sufficient to replace the application software but not the stack software. However, unless the first method is chosen, it should always be implemented as a fallback solution.

Depending on the security requirements of the application, nodes should be guarded against unauthorized replacement of the software. Corresponding security features will have to be implemented in the boot loader software.

### 6.3.2 Network Health Monitoring

WSN should implement network health monitoring to identify devices that stopped functioning and to identify maintenance needs such as battery replacement. Network health monitoring may also help in optimizing a network for reliability and performance.

Only if a coordinator node is receiving messages from another network node, it may assume that this node is functioning and is connected with the network. Every device should therefore transmit “heartbeat” messages unless there is pending data traffic. Heartbeat messages are usually empty data frames that contain at most information on battery status and other “health” information. If after a timeout period no messages arrive at the coordinator from a particular node, it can conclude that the node stopped working or has lost connection with the network and rediscovery procedures need to be started. If rediscovery fails, a broken or lost device can be signaled (for example, through the gateway to a remote operations center).

Furthermore, network health can be monitored by collecting advanced metrics using a Network Management System, with an NMS application hosted at the coordinator and NMS

agents at the leaf devices. Such metrics include the number of transmission/reception failures, of attempts to join and rejoin the network, link failures, retransmission attempts, and the checking of redundant mesh path usage. The metrics can be transmitted via a gateway into an operations center, where they will be evaluated to optimize network reliability and performance.

### **6.3.3 Broken Device Replacement**

In situations where a manufacturing facility or a building is networked using WSN, the identification of a broken device (for example, a router) may not be a trivial task. Even though the device ID of the broken device is known, it will be necessary to map the ID to the device's location. This will be easier if the installation of the devices has been documented, including the unique ID (for example, the MAC address) and the location of each device. If no documentation is available (for example, in a private installation), a visual signal will be helpful to identify a malfunctioning device (for example, a blinking LED indicating an operation error).

## **7 Summary**

Through considering three PLC phases (conception, design, service), this paper highlighted important aspects to be considered when designing a product with WSN capabilities. It differs from the regular technical literature on this topic by describing the development process of WSN based on the experience of many man years of WSN development. This information may be used by R&D managers and engineers as an incitement to solve design problems in their own implementation of WSN systems, and to judge the capabilities of suppliers that may support engineering teams in turning WSN technology into real-world solutions.

## **8 Acknowledgement**

The author would like to thank Venkatesan Venkataraman, Harish Kumar and Sunil Odayoth for their valuable suggestions and comments in formulating this paper.