**Computer Engineering**
Mekelweg 4,
2628 CD Delft
The Netherlands
**http://ce.et.tudelft.nl/**

2008

# MSc THESIS

## Reliable sensor networks

**Sander Bastiaan Kootkar**

**CE-MS-2008-03**

### Abstract

Public transport offers affordable transport to many people every day. Due to material wear down, trains can break down and stop functioning. To prevent this from happening, modern technology can be used for condition monitoring to recognize material wear down and thus allowing trains to be mended in time.
The Dutch Railways approached Logica to come up with a solution that performs condition monitoring inside a train environment. Logica's solution is the implementation of a wireless sensor network (WSN), where powered sensor nodes will be mounted as a linear array in a train compartment. These nodes will be used as a backbone network for data transport of certain train application sensor network, where data transport reliability is of major importance.
This thesis project shows how a reliable sensor network can be established. This report discusses relevant terminology on sensor networking, presents background information and quantifying models on data transport reliability and introduces relevant data transport reliability protocol solutions. Based on the train applications that will use the backbone network, this report shows what the relevant requirements are for establishing such a reliable data transport network. Based on these requirements, protocol solutions are selected that improve data transport reliability. Finally, the reliability and performance evaluation of a sensor network shows how the selected protocol solutions perform and in which way they contribute to data transport reliability.

TU Delft

Faculty of Electrical Engineering, Mathematics and Computer Science

# Reliable sensor networks
## A CASE STUDY IN COMMUTER TRAINS

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

by

Sander Bastiaan Kootkar
born in Breda, The Netherlands

Computer Engineering
Department of Electrical Engineering
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

# Reliable sensor networks

by Sander Bastiaan Kootkar

**Abstract**

Public transport offers affordable transport to many people every day. Due to material wear down, trains can break down and stop functioning. To prevent this from happening, modern technology can used be for condition monitoring to recognize material wear down and thus allowing trains to be mended in time.
The Dutch Railways approached Logica to come up with a solution that performs condition monitoring inside a train environment. Logica's solution is the implementation of a wireless sensor network (WSN), where powered sensor nodes will be mounted as a linear array in a train compartment. These nodes will be used as a backbone network for data transport of certain train application sensor network, where data transport reliability is of major importance.
This thesis project shows how a reliable sensor network can be established. This report discusses relevant terminology on sensor networking, presents background information and quantifying models on data transport reliability and introduces relevant data transport reliability protocol solutions. Based on the train applications that will use the backbone network, this report shows what the relevant requirements are for establishing such a reliable data transport network. Based on these requirements, protocol solutions are selected that improve data transport reliability. Finally, the reliability and performance evaluation of a sensor network shows how the selected protocol solutions perform and in which way they contribute to data transport reliability.

**Laboratory** : Computer Engineering
**Codenumber** : CE-MS-2008-03

**Committee Members** :

**Advisor:** Zaid Al-Ars, TU Delft

**Chairperson:** Koen Bertels, TU Delft

**Member:** Wouter Kers, Logica Rotterdam

**Member:** Steve Uhlig, TU Delft

**Member:** Almerima Jamakovic, TNO Delft

# Contents

# List of Figures

# List of Tables

# Acknowledgements

For the past 1 year and 4 months I have worked on this MSc. project with - in general - great pleasure, partially at Logica Rotterdam and partially at home in Delft. During this period I have received an enormous amount of support from the people involved in this project.
First of all, I would like to thank Zaid Al-Ars, my supervisor at Delft, University of Technology. Zaid is one of the most brightest persons I have met in my life so far. His support and advice, even when I was struggling with the project, have been a key reason to completing this project and have inspired me throughout this entire project.
Second, I would like to thank my supervisors at Logica Rotterdam, which are Wouter Kers, Dennis de Jong and Jan Benders. Each of them supported me in their own way and field of knowledge. Their interesting discussions and remarks kept me on the right track in completing this project. Even though we did not agree all of the time, I enjoyed working with all of them.
Third, I would like to thank all of the (other) Logica employees I worked with. I had a good time while working there.
Last but not least, I would like to thank my parents, my girlfriend Lonneke and all of my friends. Without their love and support, I would not have been able to complete both the Computer Engineering program and this project.

Sander Bastiaan Kootkar
Delft, The Netherlands
July 25, 2008

# 1 Introduction

**Many** people use public transport every day. Especially trains suffer from large delays. Most of the time, these delays are caused by either bad weather conditions or material breakdown. Weather conditions, of course, cannot be changed, but there is something that can be done about material breakdown.
Material that is being used so frequently as train material will eventually stop functioning. But, with the help of modern technology, material breakdown can be monitored and predicted. In this way, trains that are bound to stop functioning can be repaired in time instead of breaking down during a very crowded rush hour. This 'modern technology' is condition monitoring.
In condition monitoring several parts are equipped with sensors, which tell what e.g. the temperature, speed or vibration rate of the monitored object is. Knowing this data helps in predicting when the monitored object will fail to operate. Several applications can use this data and raise an alarm if an out-of-bound value is reached. If this is the case, the monitored part can immediately be repaired the next time the train is scheduled for maintenance.
Trains currently used by railways everyday normally have a life cycle of 30 years. Trains ordered now and running tomorrow are designed with these type of applications in mind. On the other hand, the largest part of all trains used nowadays does not have this equipment. This raises the question if these old trains should be equipped with this kind of material. Remember, refurbishing a train is a very costly operation - changes to the electrical system have to apply to strict regulations.
To solve the problem of preventing trains from breaking down, the Dutch Railways - in Dutch: Nederlandse Spoorwegen (NS) - approached Logica[1] to propose a solution to acquire condition monitoring data inside a train environment.
Logica has a large knowledge base on ICT projects inside train environments. One of Logica's departments working on this subject is Technical Software Engineering 1 (TSE1), located in Rotterdam. TSE1 has a special unit focused on sensor applications. This unit works on various projects where sensor applications are involved - one of their main interests is sensor networks. On another assignment handed out by the NS, this unit came up with an application that monitors passenger seats using a sensor network, where train passenger seats were equipped with sensor nodes to indicate the seat was taken. Each sensor node was battery-powered, so there was no problem with any regulations here. During the prototype testing of this application, the Logica engineers soon found out that the sensor nodes did not perform well inside the train environment and that test results differed for no clear reason. At one moment, they discovered that nearly all data was delivered correctly - at another moment, most data was lost. The engineers at Logica could tell it had to do with the dynamic environment the sensor network was

[1] For a more in-depth introduction to Logica, see Appendix A.1.

deployed in and the appearance of objects between the sensor nodes, but they were not able to tell what the exact cause was. Clearly, this would also be a problem for the condition monitoring assignment.
This chapter will now continue with a brief discussion on the problem statement in Section 1.1, which describes the reason for this project and discusses the central research question, that will be answered throughout this thesis report. Following this, Section 1.2 discusses the goals of the project: what are the subjects that need clarifying throughout this report? Next, Section 1.3 discusses the scope of this project, which define the boundaries of what will be discussed. This chapter concludes with an overview of this thesis in Section 1.4.

## 1.1 Problem statement

The Logica sensor application manager decided the problem of having differing data delivery test results in a train environment would be an interesting subject for graduate students to work on. He handed out a series of graduation assignments to the student graduation department of Logica, Working Tomorrow (WT)[2] to solve the problem of condition monitoring in a train environment, where the environment leads to unpredictable results.

1. The first assignment focused on the selection of the sensor nodes best suitable to this application, in which another student selected the Crossbow Mica2 sensor nodes.

2. The second assignment focused on making sure the sensor nodes could produce their own energy, so there would be no need for the nodes to be either attached to an electrical system or to use up their batteries (the principle of energy-harvesting); another student designed an example set-up.

3. The third assignment focused on deploying the sensor network inside the train environment.

In the proposed solution a single train compartment will be equipped with a number of intelligent sensor nodes, which are powered by the electrical system in the train. These nodes will have to be checked if they comply with all regulations, but when they do, these nodes can be used as a data transportation backbone network.
A number of typical train applications will use simple transmitting-only, battery-powered nodes - again, no regulations here - which are located on various positions in the train. These nodes will use the backbone node network to transport their data.
The first and second assignment were successfully fulfilled and therefor the third assignment needs to be solved. This third assignment leads to the following central research question:

*How can a reliable hybrid sensor network be established?*

The central research question is decomposed into the following partial research questions.

[2]For a more in-depth introduction to WT, see Appendix A.1.

1. *What is a reliable sensor network?*
   Chapter 2 presents the background information necessary for this project.

2. *What makes a sensor network reliable?*
   Chapter 3 analyses the applications that use the backbone network for data transport. Based on these applications a requirement analysis will me made for making a sensor network reliable.

3. *What solutions will be evaluated?*
   Chapter 4 selects the solutions that will be evaluated in Chapter 5.

4. *What are the evaluation results?*
   Chapter 5 evaluates the solutions selected in Chapter 4.

Finally, Chapter 6 summarizes this thesis, presents the project evaluation and discusses recommendations for future research.

## 1.2 Goals of project

The project goals are the following.

1. Hybrid sensor support: the sensor network must support sensor nodes that have low, transmitting-only capabilities and high, both transmitting and receiving capabilities..

2. Improve robustness sensor network: the network must be made as immune as possible to possible disruptions.

3. Reliable data transport: mechanisms must be developed to transport data with the highest reliability.

4. Network diagnosis: the sensor network must be able to diagnose itself and report problems.

5. Data encryption: the sensor network must use a simple way of encryption to protect its data.

6. Evaluation: evaluation must take place in a dynamic train environment.

## 1.3 Scope of project

The scope of the project is the boundary that tells where the project begins and ends. The boundaries were defined by Logica at the start of this project. These boundaries can best be stated when - once again - looking at the central research question:

> *How can a reliable hybrid sensor network be established?*

This section will continue with a short discussion on both the sensor network and prescribed environment scope.

### 1.3.1 Sensor network scope

The sensor network scope describes the prescribed components used in this project. Figure 1.1 shows an overview of all components involved in this project, with in Figure 1.1(a) [6] the relationship between the different components and in Figure 1.1(b) [6] the corresponding legend.


(a) Relationship different components [6] (b) Legend [6]

Figure 1.1: Sensor network scope

The legend in Figure 1.1(b) shows the following components:

1. The sensor nodes are the basic network components and are already available. Section 1.1 stated that the selected sensor nodes are the Crossbow Mica2 sensor nodes[3] which operate on running TinyOS[4]. As a result of this, several nodes and one programming board were bought by Logica. 8 of these nodes are available and these nodes have to be used in this project.
   Even though the nodes are already available, the way they connect to each other and the specific way they behave with respect to data transport is yet undefined.

2. The gateway consists of a node connected to a Printed Circuit Board (PCB). This item is commonly referred to as the gateway node or gateway board and is used for acquiring data received and processed by the node connected to the PCB. The gateway node is already available.

3. Both the PC and remote server are used for interfacing the gateway node to acquire data. A server and special application is available that collects data from the Crossbow Mica2 programming board and displays this in a console-window.

### 1.3.2 Prescribed environment scope

All experiments must be conducted in a dynamic train environment. This is a single train compartment, as was stated in Section 1.1. This environment has the following properties.

[3] Crossbow Mica2 sensor nodes are discussed more in-depth in Appendix A.2.
[4] For a description of programs and files used, see Appendix A.3.

1. The data transportation backbone will have a linear array topology. The size of the linear array will vary, but up to a maximum number of 8 nodes.

2. For simplicity reasons, the applications that use the transportation backbone are represented as one single node, which is the same Crossbow2 Mica2 sensor node as the nodes used in the backbone configuration.

## 1.4 Overview of thesis

The lay-out of this thesis is as follows. First, Chapter 2 introduces the background information, necessary for understanding the context of this project. Following this, Chapter 3 discusses the requirements for implementing a sensor network in the train environment. Chapter 4 discusses the solutions that will be evaluated to create a reliable sensor network. Following this, Chapter 5 shows the reliability and performance evaluation of the sensor network. Finally, Chapter 6 summarized the conclusions that can be drawn from this project, presents the project evaluation and ends with recommendations for further research.

# Background information 2

This chapter provides the required background knowledge needed to understand this project. First, the terminology related to hybrid wireless sensor networks is introduced in Section 2.1. Following this, Section 2.2 introduces what reliability means in sensor networking and in which way this can be modeled. Third, Section 2.3 discusses the various protocols available that focus on improving reliability on the OSI layers applicable to this project[1]. This chapter ends with a conclusions that can be drawn from the background information in Section 2.4.

## 2.1 Hybrid wireless sensor network terminology

This section discusses the terminology related to wireless sensor networks to provide an introduction into the principles and background of wireless sensor networking. The definition of a hybrid wireless sensor network is:

> *A network, built from hybrid wireless sensor nodes, which are deployed and operate decentralized and self-organizing.*

### 2.1.1 Hybrid wireless sensor nodes

Hybrid sensor nodes are the small, wireless and light-weighted embedded devices that, together, create a wireless[2] sensor network. These nodes can be used in various applications, such as measuring temperatures or vibrations.
Hybrid refers to the type of nodes used; in a hybrid sensor network, both high-capability and low-capability nodes are used.

1. High-capability nodes can both transmit and receive data. An example high-capability node is the Mica2 nodes shown in Figure 2.1 [6].

2. Low-capability nodes can only transmit data.

Figure 2.1: Mica2 sensor node [6]

Data is transmitted by electromagnetic waves.
Nodes have a limited size and weight; for example, the Crossbow Mica2 displayed in

[1]The OSI model and the layers applicable to this project are discussed in detail in Appendix A.4.
[2]In this context, wireless means there are no physical connections (e.g. cables) between the sensors.

Figure 2.1 has a size of $58 \times 32 \times 7$ millimeter (length $\times$ width $\times$ height) and weighs only 18 grams. With new (and smaller) technology, these numbers are likely to decrease even more in the future. As a result of these small sizes, the number and sizes of the components on the node are small.

#### 2.1.1.1 Node components

The most common node components are the Central Processing Unit (CPU), memory, transceiver, sensor and power source.

1. The CPU or microprocessor is the 'heart' of the sensor node. It provides the computational power for arithmetic and logical computations.

2. Sensor nodes are equipped with both Read-Only-Memory (ROM) and Random Access Memory (RAM) for storage of the processed data and programs.

3. A transceiver is used for gaining access to the shared physical medium and for transmitting and receiving data to and from other sensor nodes.
   The shared physical medium in this case is a radio channel, which operates on a certain frequency. This frequency depends on which channel is approved in which country for transmission of data. Access to the shared physical medium is gained according to a specific Medium Access Control (MAC) protocol.

4. A sensor is used for observing and converting the monitored property (e.g. temperature or vibrations) into a digital signal. This digital signal can then be processed by a sensor node and the wireless sensor network.

5. Sensor nodes need energy for functioning. Energy can be extracted from a small attached battery or by energy harvesting [17]. In energy-harvesting, the energy is extracted from the node's environment and saved in e.g. a capacitor for later use.

Most nodes run an operating system, which is specifically designed for wireless sensors. One of the most commonly used wireless sensor operating systems is TinyOS [25][3].

#### 2.1.1.2 Node limitations

Nodes have specific limitations that limit the functionality of the sensor network. Implementing more functionality will expand the possibilities of the nodes but will use more resources. Typical node limitations are low bandwidth, energy-usage and low computational capacity.

1. Sensor nodes generate traffic. When data is aggregated in some part of the network, it could happen that the traffic load is too high and the maximum bandwidth threshold has been passed. The network will start losing messages, simply because the sensor nodes cannot cope with the amount of data any more.

[3] For a description of programs and files used, see Appendix A.3.

2. Some of the sensor nodes used are not powered by wires but use energy from batteries or energy harvesting [17]. If a node is not provided with enough energy in time, it will stop functioning when it does not have enough energy left.

3. Since nodes are often small, they have relatively slow processors and a little amount of memory available. The computational capacity they provide is limited.

### 2.1.2 Deployment

*Deployment is the problem of determining where the sensor nodes are placed.*

The way the sensor nodes are deployed defines what the the topology of the network is. Various definitions and types of the term 'topology' are used. This section discusses the definitions and types used in this report.

#### 2.1.2.1 Definition topology

*Topology is the mapping or lay-out of the sensor nodes.*

Topology can be seen as a map, showing the physical placing of the sensor nodes with a line or link between some of them. The link indicates there is a connection between two nodes: the nodes can reach each other when transmitting data. A difference exists in terminology between the physical and network topology.

1. The physical topology is purely the place - or coordinates - the sensor nodes have on a map.

2. The network topology shows this map, with lines indicating that the involved nodes have a connection.

#### 2.1.2.2 Topology types

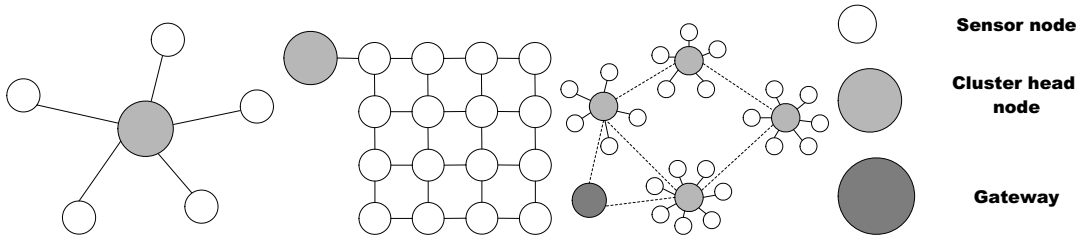
Various topology types exist, such as single-hop, multi-hop and hybrid topologies.

1. In a single-hop configuration, the gateway node is reached in one single step. An example topology is the star, which is on display in Figure 2.2(a) [10].

2. In a multi-hop configuration, the gateway node is reached in multiple steps. An example topology is the mesh, which is shown in Figure 2.2(b) [10].

3. In a cluster configuration, the single- and multi-hop principle is combined to form a combination of these two. In this case, local gateway nodes are assigned, so-called cluster heads. An example topology is shown in Figure 2.2(c) [10].

Topology information is absolutely necessary for sending data to a certain destination node.




(a) Single-hop: star [10] (b) Multi-hop: mesh [10] (c) Cluster [10] (d) Legend [10]

Figure 2.2: Example topology types

### 2.1.3 Decentralization

> *Decentralization is the process of dispersing decision-making closer to the point of service or action* [1].

The sensor nodes are the point where the "action" (the measuring of a certain property) takes place and they decide what to do with the data. The information on what to do with the data originates from the applications that run on the sensor node. These applications initiate a new connection and transmit the data to a destination they themselves have defined.

### 2.1.4 Self-organizing functionality

Self-organizing functionality is the ability of the sensor network to initialize its topology, from the very first moment after it has been deployed.
The initial topology is the topology of the sensor network when it first starts functioning. The nodes need information on the topology if they want to route data to it. Therefore, the sensor network must know what the initial topology is. If the initial topology is not known, the network must find out what the topology is by itself. This initial topology is either predefined or ad-hoc.

1. In an ad-hoc deployment, the location of the nodes is not known on forehand.

2. If the deployment is predefined, the location of the nodes is not known on forehand.

## 2.2 Data transport reliability

This section discusses data transport reliability in sensor networking. Reliability can have a very wide and abstract meaning. Therefore, the definition is narrowed down: from now on, the meaning of reliability depends on the commonly used metrics of reliability in wireless sensor networking.
With the metrics as a starting point, the ideal model of reliability will be investigated. Naturally, there are various downsides to this model, but it serves as a good starting point to introduce radio irregularity.

Much research has gone into finding the reasons for radio irregularity - this will eventually help to quantify radio irregularity and model practical results.

### 2.2.1 Metrics used

The most commonly used metrics when it comes to calculating reliability in sensor networking are Packet Reception Rate (PRR) and Received Signal Strength Indicator (RSSI) and their relationship to distance.

#### 2.2.1.1 Packet Reception Rate

The Packet Reception Rate (PRR) is calculated as the ratio between received and sent packets; it is measured as the percentage of packets that arrived undamaged on a link, or (in formula):

**Theorem 2.1**

$$\text{Packet Reception Rate } = \frac{\text{\# packets received by node}}{\text{\# packets sent to node}}$$

#### 2.2.1.2 Received Signal Strength Indicator

The Received Signal Strength Indicator (RSSI) is a measurement of the power in a received radio signal. Quite typically, in sensor nodes, this value is sampled by an A/D converter. When the properties of the A/D converter are known, the sampled value can be used to calculate the signal strength. The Received Signal Strength Indicator (RSSI) is calculated as the output power of the transmitting node minus path loss:

**Theorem 2.2**

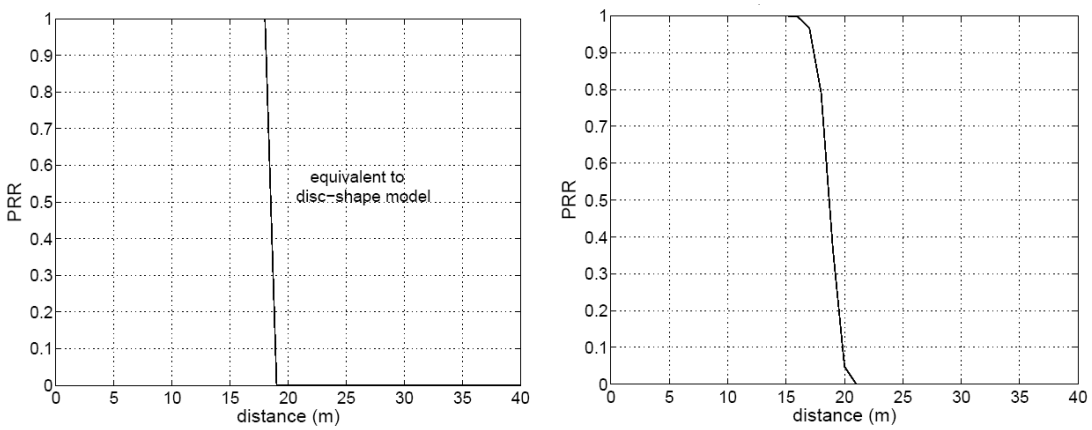
$$RSSI(d) = P_t - PL(d)$$

where

1. $RSSI$: the signal strength of the received radio signal at distance $d$ in dB;
2. $P_t$: the node's transmitting power in dB;
3. $PL(d)$: the path loss in dB at distance $d$;

### 2.2.2 Ideal model

In an ideal world, the transmission range of a node would be equivalent to a disc-shape model, just like the one on display in Figure 2.3(a) [4]. In this model, a receiving node would be able to receive all data from this transmitting node correctly up to a certain range, which is 18 meters in this figure. Obviously, this does not make very much sense, since this world is not an ideal world. One of the intervening factors is the node itself. Figure 2.3(b) [4] shows the model from Figure 2.3(a), where the node has a slightly more realistic transmission range - up to 15 meters, the PRR is 100 % but from 15 to 18 meters it drops down in an almost linear way to 0 %. Though slightly better, this improved




(a) Ideal environment - ideal transceiver [4] (b) Ideal environment - real transceiver [4]

Figure 2.3: PRR versus distance for different scenarios

model turns out to be far from the truth, as will be proved in the following sections. Summarizing, 2 different regions can be clearly seen in the previous figures: the connected region and the disconnected region.

1. In the connected region there is a PRR of 100 %.
2. In the disconnected region the PRR is equal to 0 %.

### 2.2.3 Radio irregularity

Many practical tests prove that in real world applications the models described in Section 2.2.2 are not (completely) correct. Experiments show there is a so-called 'radio irregularity', which causes a irregularity in radio range and variations in packet loss at a given distance from the transmitter. Many research has gone into proving and calculating the outcomes of experiments, which had to do with radio irregularity.

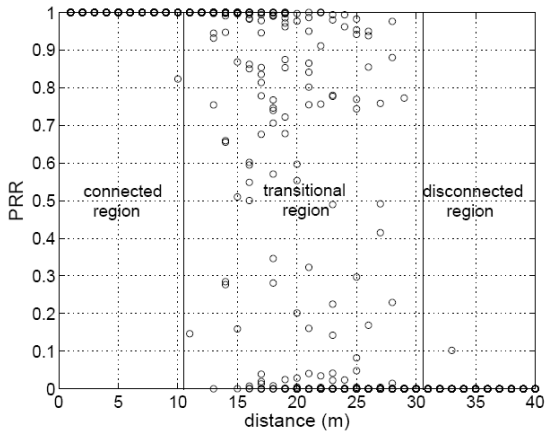
#### 2.2.3.1 Transitional region

Most striking difference when looking at the real world observations is there appears to be a so-called third region, the 'transitional region'.
Figure 2.4 shows the results of an experiment conducted in [4], where researchers investigated the real relation between signal strength, PRR and distance. They found out there was a third region, the 'transitional region' in which it is not clear what the PRR of the receiving node is. The same type of experiments with the same results - being, the existence of this transitional region - were conducted in [1, 4]. Summarizing, there appear to be 3 regions in a node-to-node link.

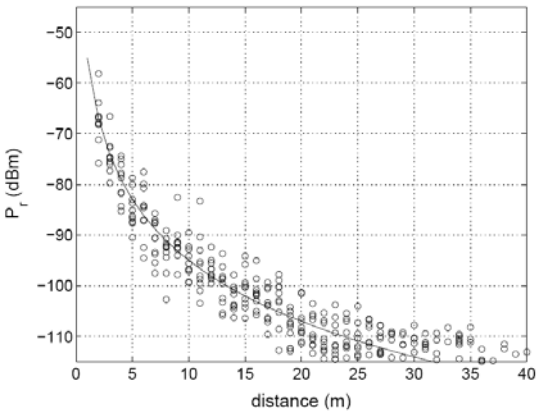
1. Connected region: region with 100 % PRR.
2. Transitional region: region with unpredictable PRR.

3. Disconnected region: region with 0 % PRR.

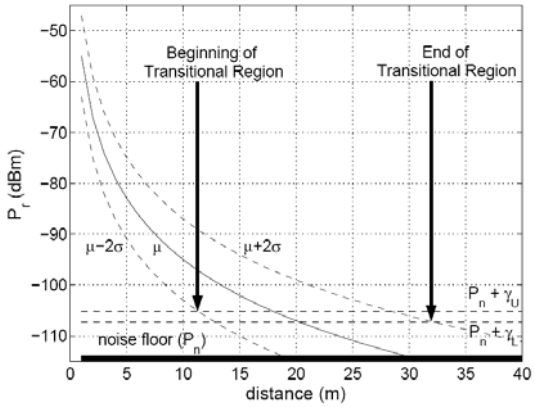
This troubles things, due to the transitional region. Fortunately, there is a way to predict and calculate when these regions appear. This method will be explained in Section 2.2.5.




(a) PRR versus distance from experiments [4].




(b) Signal strength versus distance from experiments [4].




(c) Figure 2.4(b), now indicating the transitional region [4].

Figure 2.4: Real world observation results

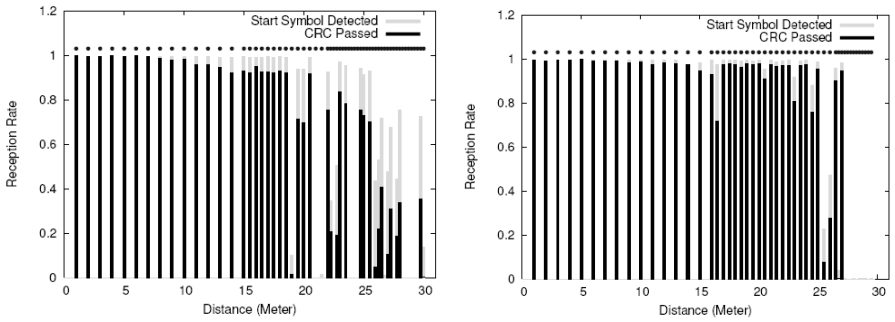
#### 2.2.3.2 Environment-dependence

The authors of [1] performed experiments in three different environments using Mica sensor nodes:[4]

[4]It is not clear whether or not these are the same nodes as the Mica2 nodes at Logica, this is not described in the experiment results.

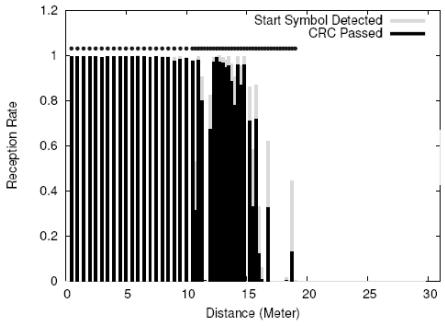
1. An office building; office buildings are particularly harsh wireless environments, due to the likelihood of multi-path reflections from the walls and interference from other obstacles.
2. A car park with no obstacles.
3. A local state park - including trees;

One of the experiments' outcome was a profile of the PRR over distance for each of the three described environments - this shows some interesting results. Figure 2.5 shows the experiment results from the experiment conducted in [1]. Here, 'start symbol detected' means the start symbol of the data packet has been recognized and 'CRC passed' means the data packet has passed the Cyclic Redundancy Check (CRC) - meaning, it is a valid data packet. Looking at these figures, it can be stated that even in the most harsh




(a) Spatial profile for office environment [1]

(b) Spatial profile for car park environment [1]




(c) Spatial profile for state park environment [1]

Figure 2.5: Spatial profile for 3 different environments

environment - the office environment in Figure 2.5(a) [1] - packet reception rates are still high. The difference lies in the gray areas. Besides this, the width of the gray area is interesting as well - in the office environment, it is one third, while in the park, it is only one fifth. While obvious, it is now clear that environments largely influence the spatial profile of a transmitting node's transmission range.

#### 2.2.3.3 Coding schemes

Another interesting outcome of the experiment conducted in [1] was the variance in packet loss over different coding schemes used. The following coding schemes were used:

1. 4-bit / 6-bit (4B6B) coding encodes one byte (8 bits) into 12 bits, with the capability of detecting 1 bit error out of 6 bits.
2. Single Error Correction, Double Error Detection (SECDED) coding encodes each byte into 24 bits - this is the default Crossbow Mica2 TinyOS encoding.
3. Manchester encoding encodes each byte into 16 bits, with the capability of detecting an erroneous bit out of 2 bits.

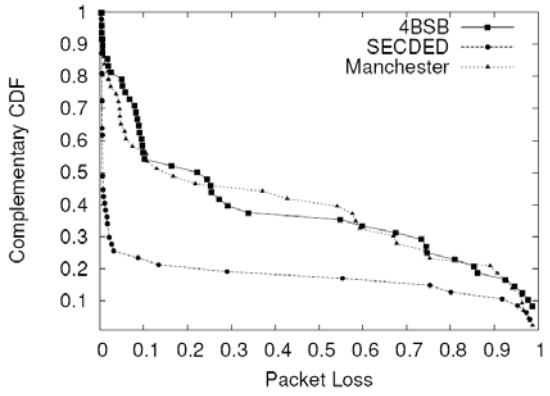



Figure 2.6: Distribution of nodes over packet loss for different coding schemes [1]

Figure 2.6 [1] shows the results. 4-bit / 6-bit is the least error tolerant, followed by Manchester and Single Error Correction, Double Error Detection.

#### 2.2.3.4 Temporal correlation

The third and perhaps most surprising outcome of the experiment in [1] is the following: packet delivery performance can differ over time - and not just a few percent! The experiment was conducted over a 2 hour window with a 40 second interval, and a receiver near the edge of the transmission range can have a PRR that varies between 20 to 60 % - see Figure 2.7 [1].

#### 2.2.3.5 Non-isotropic signal strength

Another interesting experiment was conducted in [3], which shows some other interesting results. The first one is that, apparently, the power with which the transceiver is transmitting its data is not consistent in each direction. A single node was programmed to transmit data to nodes in 4 different directions at the same distance. The results are shown in Figure 2.8 [3], where 'Beacon SeqNo' on the x-axis means the number of

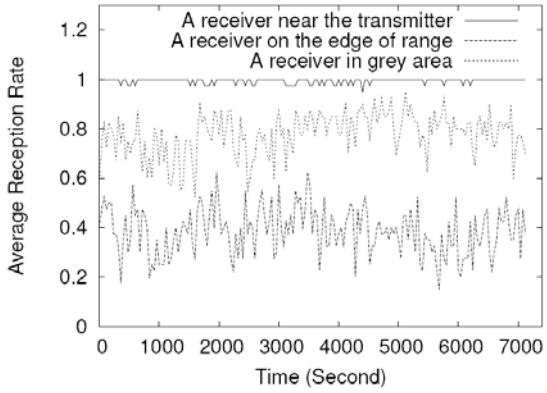



Figure 2.7: PRR change over time [1]

transmitted data or beacon packets. The y-axis shows the corresponding RSSI values. Even though the received signal strength is stable over time, the signal strength received in the south is much higher than the one received in the east. Another outcome of the

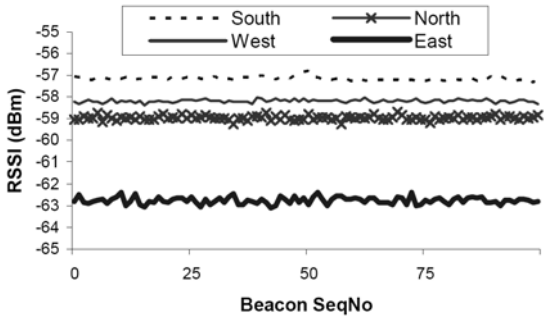



Figure 2.8: Signal strength over time in four different regions [3]

experiment described in [3] is rather obvious - when the received signal strength varies in different regions, so does the packet loss ratio.

### 2.2.4 Reasons for irregularity

So far, several experiments have been discussed, with results not matching what intuitively could be expected. This can be best explained by looking at the components used and the propagation media the experiment takes place in.

#### 2.2.4.1 Components used

The components used here are the components on the sensor node. The sensor node's antenna and transceiver have a large influence on the transmission range and direction.

1. The antenna type can be either directional or omni-directional.

   (a) A directional antenna radiates greater power in one or more directions.
   (b) An omni-directional antenna directs power uniformly in one plane with a directive patten shape in a perpendicular plane. The only 3 dimensional omni-directional antenna is the unity gain isotropic antenna - see [14].

2. The transmission power is another major asset. Due to an incorrect hardware calibration - or simply, due to the way the antenna is designed - the node may not have the same antenna gain along all directions. This is clearly what led to the results in Section 2.2.3.5.
   Besides this, if the nodes are battery-powered - which is not the case in the node backbone application, but in e.g. the seat occupancy application - there might be another problem - the transmitting power depends on the power source and external battery power level may decrease with different rates.

3. The receiver sensitivity defines the receiver's ability to correctly receive data packets.

4. If the receiver has a low threshold, the receiver is more likely to receive data. Whether or not this data is correct - or not is a different story.

5. The transceiver's noise figure defines if the transceiver is able to receive data which has a large SNR - or not.

#### 2.2.4.2 Propagation media

Besides the components used, an equally large influence is the propagation media the application takes place in.

1. The media type the signal propagates in is - again - largely influential. To travel from the transmitting node to another destination - e.g. a receiving node - the signal propagates within a medium. Electromagnetic waves are propagated better through air than water. Other materials, e.g. concrete, are even worse and almost impossible to propagate through. In the medium, it can either be reflected, diffracted and scattered [27].

   (a) Reflection occurs when an object is encountered that is larger than the signal's wavelength.
   (b) Diffraction occurs when an irregular structure is encountered.
   (c) Scattering occurs when the medium through which the signal propagates contains objects, smaller than the signal's wavelength.

2. The background noise must be as low as possible, otherwise the transceiver is not able to distinguish noise from the actual transmitted data.

3. Last, but not least, are the environmental factors.

   (a) The temperature of the propagation medium limits - or expands - the range of the transmitted signal.
   (b) Obstacles in the way of the propagated signal will cause the signal to divert.

### 2.2.5 Modeling practical results

Many strange things happen in sensor node applications - at first sight, producing even rather unpredictable results. This section will be used to present theoretical models to explain these practical results. By using the simple metrics presented in Section 2.2.1, these models will provide valuable information on the expected PRR and the starting and ending point of the transitional region in a certain environment. Besides this, it will introduce the path loss exponent $\eta$, which is a metric used to indicate the amount of deterioration in signal strength in a specific environment. Therefore, it is very useful as a metric.
Previous research has been able to extract formulas which provide this valuable information. Three separate theoretical models will be introduced, which put together provide the information on the PRR and transitional region. The starting point for these calculations is the Radio Frequency propagation model in Section 2.2.5.1, which describes in which way the signal strength of a transmitted data packet deteriorates over distance. Second of all, the PRR model is described in Section 2.2.5.2, which describes what the expected PRR is at a specific distance. Third, the noise floor model is discussed in Section 2.2.5.3, which describes the theoretical model on how to estimate the noise floor level in a specific environment. Finally, Section 2.2.5.4 adds all these 3 elements together.

#### 2.2.5.1 RF propagation model

The most commonly used radio propagation model is the log-normal shadowing path loss model [5]:

**Theorem 2.3**

$$PL(d) = PL(d_0) + 10\eta \log_{10}(\frac{d}{d_0}) + \sigma$$

where:

1. $PL(d)$: the path loss in decibel (dB) at distance d;
2. $PL(d_0)$: the reference path loss at distance $d_0$;
3. $\eta$: path loss exponent for the environment that is measured - for a 'normal' environment such as an outside park, the path loss exponent lies between 2 and 4; for a 'difficult' environment such as an office room, the path loss exponent lies between 4 and 6;
4. $\sigma$: a zero-mean Gaussian Random Variable with standard deviation $\sigma$ in dB;

#### 2.2.5.2 Radio reception model

The relationship between the packet reception rate $p$ and the Signal-to-Noise Rate $\gamma$, as defined in [4], is:

**Theorem 2.4**

$$p = (1 - \frac{1}{2}\exp^{-\frac{\gamma}{2}\frac{1}{0.64}})^{16f}$$

where:

1. $p$: the probability of successfully receiving a packet, using Manchester encoding - also called Packet Reception Rate (PRR)[5];
2. $\gamma$: the Signal-to-Noise Rate (SNR);
3. $f$: the frame size in bytes[6];

Figure 2.9 [4] shows Theorem 2.4 for a frame size of 50 bytes.

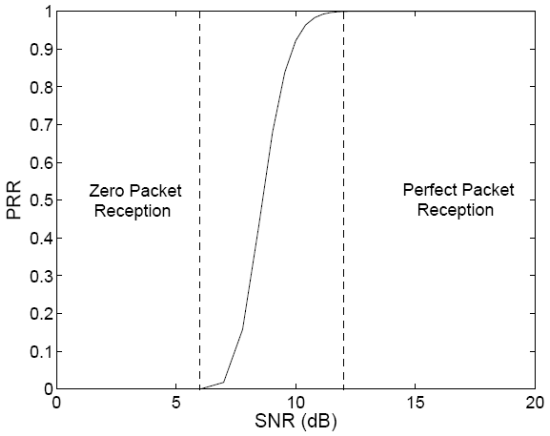



Figure 2.9: Radio reception model [4]

#### 2.2.5.3 Noise

The noise floor can be calculated using the following theorem [4]:

**Theorem 2.5**

$$P_n = (F + 1)kT_0B$$

where:

1. $P_n$: the noise floor level in dB;
2. F: the node's noise figure in dB;
3. $k$: Boltzmann's constant: $1.3806503 \times 10^{-23} J/\circ K$;
4. $T_0$: the ambient temperature;
5. B: the node's bandwidth;

For example, the Mica2 sensor nodes have a noise figure of 13 dB and a noise bandwidth of 20 kHz [6]. In an environment that has an ambient temperature of 300° Kelvin (equivalent to 27° Celsius), the noise floor is -118 dBm.

[5]Manchester encoding is the (only) encoding method supplied with TinyOS v1.15.

[6]A frame consists of a preamble, the data packet itself and the Cyclic Redundancy Check (CRC).

#### 2.2.5.4 All together

The Signal-to-Noise Rate can be calculated using the following formula [4]:

**Theorem 2.6**

$$\gamma(d) = P_t - PL(d) - P_n$$

where:

1. $\gamma(d)_{dB}$: the SNR at distance $d$;
2. $P_t$: the node's transmitting power in dB;
3. $PL(d)$: the path loss in dB at distance $d$;
4. $P_n$: the noise floor level in dB;

On the other hand, most modern transceivers (such as the CC1000 transceiver, which is used on the Mica2 node) provide the Received Signal Strength Indicator (RSSI) value, which can also be perfectly used in this computation. Theorem 2.2 showed how to derive this RSSI value.

$$RSSI(d) = P_t - PL(d)$$

where

1. $RSSI$: the signal strength of the received radio signal at distance $d$ in dB;
2. $P_t$: the node's transmitting power in dB;
3. $PL(d)$: the path loss in dB at distance $d$;

When combining Theorem 2.6 and Theorem 2.2, this provides the SNR while using the RSSI indicator:

**Theorem 2.7**

$$\gamma(d) = RSSI(d) - P_n$$

where:

1. $\gamma(d)_{dB}$: the SNR at distance $d$;
2. $RSSI$: the signal strength of the received radio signal at distance $d$ in dB;
3. $P_n$: the noise floor level in dB;

The formula from Theorem 2.4 provided the PRR at a distance $d$. Now that the SNR is known, the PRR can be calculated using the following formula [4]:

**Theorem 2.8**

$$p(d) = (1 - \frac{1}{2}\exp^{-\frac{\gamma(d)}{2}\frac{1}{0.64}})^{16f}$$

where

1. $p$: the probability of successfully receiving a packet, using Manchester encoding at distance $d$;
2. $\gamma(d)$: the SNR at distance $d$;
3. $f$: the frame size in bytes - see comments in Footnote 6.

The following formula gives the relationship between a certain PRR and its corresponding SNR [4].

**Theorem 2.9**

$$\gamma = 10\log_{10}(-1.28\ln(2(1 - PRR^{\frac{1}{16f}})))$$

where

1. $\gamma$: the SNR for the corresponding PRR;
2. $PRR$: the Packet Reception Rate;

Finally, this formula is used to derive both the starting point $d_s$ and ending point $d_e$ of the transitional region. The starting point $d_s$ is the following [4]:

**Theorem 2.10**

$$d_s = 10^{(\frac{P_n + \gamma_u - P_t + PL(d_0) + 2\sigma}{-10\eta})}$$

The ending point $d_e$ is the following:

**Theorem 2.11**

$$d_e = 10^{(\frac{P_n + \gamma_l - P_t + PL(d_0) - 2\sigma}{-10\eta})}$$

where

1. $P_n$: the noise floor level in dB;
2. $\gamma_u$: the corresponding SNR for PRR $u$ - see Theorem 2.9; this is the SNR for the PRR, defined as the start of the transitional region. For example, if the transitional region is defined to start below PRR = 0.9, this value is equal to 0.9.
3. $\gamma_l$: the corresponding SNR for PRR $l$ - see Theorem 2.9; this is the SNR for the PRR, defined as the end of the transitional region. For example, if the transitional region is defined to end below PRR = 0.1, this value is equal to 0.1.
4. $P_t$: the node's transmitting power in dB;
5. $PL(d_0)$: the reference path loss at distance $d_0$;
6. $\sigma$: Standard deviation value;
7. $\eta$: path loss exponent for the environment that is measured;

By knowing these formulas, the minimal signal strength value (and thus the distance) can be calculated for which the node is very likely to have a good reception rate - thus, without being in the transitional region in which unpredictable events happen. This knowledge is very useful when it comes to placing the nodes correctly. This knowledge will be used and evaluated in Chapter 5.

## 2.3 Relevant OSI layer solutions

This section discusses the solutions available that focus on reliability - with respect to this project, that is. Therefor, the OSI model is used. The OSI model is a layered, abstract description for communications and computer network protocol design, consisting of seven layers. Please refer to Appendix A.4 for further details on the OSI model. The layers applicable to this project are Layer 2, the data link layer and Layer 3, the network layer. Therefor, the following sections will be used to discuss reliability solutions on both of these layers. First, Section 2.3.2 discusses the data link layer. Second, section 2.3.1 discusses the network layer.

### 2.3.1 Network layer

This section discusses the various available reliability solutions, with respect to the network layer. The functionality of the network layer is the following, as is stated in Section A.4.1.1:

> *The network layer provides mechanisms for routing the data units across the network.*

Routing provides the mechanisms necessary to deliver a data packet at the correct recipient. Routing is one of the research topics that has gained a lot of interest, which means that many, many routing protocols exist. In general, a data packet is equipped with an address, that determines the destination of the data packet. The most commonly known routing types are metric-based routing, multi-path routing, lifetime maximizing routing, geographic routing, source routing and mobility routing. This section's aim is not to discuss all routing principles and routing protocols but to explain the principle, together with an example of a protocol.

#### 2.3.1.1 Metric-based routing

In metric-based routing, the node decides which node to send its data to based on a metric. The metric tells which node is best suitable - for example, with link has the highest likeliness of reception or which node is nearest. Advantage of this method is that the node can decide to use another path in case the first option fails (for example, if there is no acknowledgment in a certain amount of time). Again, topology knowledge is necessary and must not change. Another downside is that each node must keep track of a so-called neighbor table, containing the level of quality of all nodes reachable. This will consume some memory, but this is not the largest downside: to fill the neighbor table with data, it needs to either just 'transmit and see' - just send data to the first node, see if it arrives and if not, try another one - to gain statistics or before setting up a connection, start broadcasting.
Another goal of metric based routing can be lifetime saving. An example protocol is described in [16], in which the authors define a metric to achieve this goal.

#### 2.3.1.2 Multi-path routing

Multi-path routing uses the principle of using multiple paths to reach its destination - under the simple assumption that the likeliness that a data packet reaches its destination is higher using multiple paths instead of one single path. For example, the braided multi-path [15] protocol uses a 'main path' and maintains several alternate paths, to be used when the main path fails. This method indeed secures a higher chance of delivering data at the recipient and seems especially useful when battery-powered nodes are used; if these nodes fail, the multi-path protocols will simply use another path.
On the other hand, this introduces a new problem: besides setting up the main path and the neighbor table necessary for the main path, it needs to do the same for the alternate paths. This will introduce an even higher overhead in time and memory costs to keep the network functioning.

#### 2.3.1.3 Static routing

Static routing is perhaps the most simple way of routing. In static routing nodes are statically programmed to forward data to the next node.

#### 2.3.1.4 Geographic routing

In geographic routing all nodes have knowledge of each other's position before the initial transmission starts. The simplest way is to just forward the data packet to the closest neighbor, as described in [11]. This requires little programming. If the data does not reach the intended recipient, there is no way for the transmitting node to find out and try again in another way.

#### 2.3.1.5 Source routing

Source routing creates a route on-demand when a transmitting computer requests one. However, it uses source routing instead of relying on a routing table at each intermediate device, which means that the source decides what the route is. Source must be seen wider than just the initiator of the route; every intermediate node is the source of the rest of the route and thus decides what the following hop is. An example protocol is the Dynamic Source Routing (DSR) protocol [13], which uses the source routing principle and consists of two phases: route discovery and route maintenance.

#### 2.3.1.6 Mobility routing

A special case are moving nodes, which demands a highly-dynamic protocol. This is outside the scope of this project, since the backbone nodes in the backbone application have a fixed position.

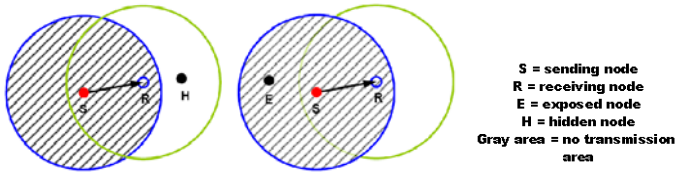
### 2.3.2 Data link layer

This section discusses the various available reliability solutions, with respect to the data link layer. The functionality of the data link layer is the following, as is stated in Section

A.4.1.2:

> *The data link layer provides mechanisms for accessing the shared physical medium by the device.*

Accessing a shared physical medium is the objective of the Medium Access Control (MAC). The Medium Access Control mechanism controls the access of a node to the shared medium and must make sure that only one node at a time controls the shared medium. A typical problem that can arise is the hidden and exposed node problem, which are shown in Figure 2.10.




(a) Hidden node [9] (b) Exposed node [9] (c) Legend [9]

Figure 2.10: Hidden and exposed node problem

1. In the hidden node problem [7, 21, 22] a node S is transmitting to node R. Node H, which is outside the transmission range of both node S, will see that the channel is clear and can start transmitting to node H. S and H are hidden to each other and the collision cannot be avoided by one of the basic protocols. Figure 2.10(a) [7] illustrates this problem.

2. In the exposed node problem [7, 22] node R is transmitting data to node S, whereas node H wants to transmit data to node E. Node H is in the range of R and will sense that the channel is occupied and will not send its data. Node E is outside R's range and S is outside of H's range, so it should not be a problem for both of them to transmit data at the same time. Figure 2.10(b) [7] illustrates this problem.

MAC protocols can be divided into three subcategories, the so-called MAC classes. The devision in these 3 classes was introduced in [9] and allows for a very clear categorization.

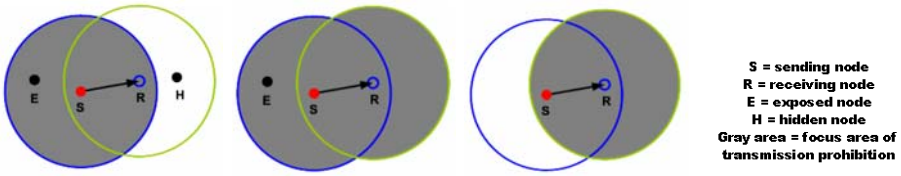
1. The MAC class 1 category of protocols exist of protocols that only control medium access by prohibiting channel access within the sender's radio range. Thus, the protocol still suffers from the hidden and exposed node problem. This can be clearly seen in Figure 2.11(a) [9]; nodes S and R are in each other's region and in this specific region, they can find out if another node is trying to gain access. On the other hand, node H is still hidden to node S and node E is still exposed to node S.

2. The MAC class 2 category of protocols exist of protocols that prohibits simultaneous channel access within the sender's and receiver's radio range. By doing this, it prevents the hidden node H from gaining access while node R is transmitting.

In this way, the hidden node problem is solved. This situation is shown in Figure 2.11(b) [9].

3. The MAC class 3 category of protocols exist of protocols that prohibits simultaneous transmissions within the receiver's radio range and simultaneous transmissions toward nodes within the sender's radio range. Again, by prohibiting simultaneous transmissions within the receiver's radio range it prevents the hidden node problem. If a node now wants to send to a node outside the region of node S, it is prohibited channel access as well and thus solves the exposed node problem; see Figure 2.11(c) [9].

For each MAC protocol class Table 2.1 [9] shows whether or not nodes in the indicated region (which refer to Figure 2.11) may send or receive. If there is extra comment available, the index number is stated in parentheses. Here is the comment from Table




(a) Class 1 [9] (b) Class 2 [9] (c) Class 3 [9] (d) Legend [9]

Figure 2.11: MAC protocol classes

| | Nodes in E region | | Nodes in H region | | Nodes in S-R region | |
|---|---|---|---|---|---|---|
| May: | Send | Receive | Send | Receive | Send | Receive |
| MAC class 1 | No (1) | Yes (2) | Yes (3) | Yes (4) | No | Yes (2) |
| MAC class 2 | No (1) | No (5) | No | No | No | No |
| MAC class 3 | Yes (6) | No | No | Yes (7) | No | No |

Table 2.1: MAC classification [9]

2.1, as was discussed in [9].

1. The exposed node problem; if the destination node is outside the R's coverage range, this would have been allowed.

2. If the transmitting node is outside the range of S, the transmitting node is allowed to send.

3. Node may transmit to any node.

4. Node may receive from every node outside node S.

5. No, since if E was allowed to receive data, it would collide with data transmitted by S.

6. Allowed to transmit to nodes outside the transmission range of node R, which solves the exposed node problem.

7. Allowed if the transmitting node is outside the range of node R.

This section continues with an overview of the several different MAC protocols available in Class 1, Class 2 and Class 3.

#### 2.3.2.1 Class 1 MAC protocol

The most common class 1 protocols are Aloha and Carrier Sense Medium Access (CSMA). Both protocols are contention-based protocols. Contention is a condition that arises when two or more nodes attempt to transmit at the same time over a shared channel. Since they are in class 1, these two protocols still suffer from the hidden and exposed node problem.

1. Aloha is one of the very first medium access protocols implemented. Aloha comes in two versions, unslotted and slotted Aloha.
   The principle of unslotted Aloha [18, 19] is to send data, if there is any data to be send. If there is a collision with another transmission (meaning that another node is already transmitting data), the data must be retransmitted later, after a random period.
   Slotted Aloha [19] uses the same principle, but it will only allow retransmissions in specified slots.

2. In CSMA [20] a node listens to the channel first if it is clear. Pure CSMA does this by detecting the carrier wave to avoid collisions. This means that if two nodes try to send data at nearly the same moment, neither of them will detect a carrier and will start transmitting. As a result, energy is wasted since the complete data frame is transmitted. More advanced versions of CSMA are Carrier Sense Medium Access with Collision Avoidance (CSMA/CA) and Carrier Sense Medium Access with Collision Detection (CSMA/CD).
   In Carrier Sense Medium Access with Collision Avoidance [20], each node listens to the channel to see if it's free - this is not the same as just checking for the carrier wave, as does CSMA. If the channel is free, the node will start its transmission. Collisions can still occur. If the channel is not free, it will wait for a random amount of time before it will try to access the channel.
   In Carrier Sense Medium Access with Collision Detection [23, 24], sending nodes are able to detect when a collision occurs and stop transmitting immediately, backing off for a random amount of time before trying again. However, it is not possible with all media such as radio.

#### 2.3.2.2 Class 2 MAC protocol

Class 2 protocols provide a solution to the hidden node problem. Example protocols are Sensor-MAC (S-MAC) and Carrier Sense Medium Access with Collision Avoidance with reservation.

1. Sensor-MAC [29] aims at energy-efficiency by avoiding idle listening. S-MAC is a combination of a contention-based and Time Division Multiple Access (TDMA) protocol. In TDMA, nodes share the same frequency channel by dividing the signal into different time slots. Each node has its own time slot. Downside here is that the slot assigning has to be synchronized among the available nodes. On the other had, the protocol saves energy and increases reliability by avoiding idle listening and avoiding hidden node collisions.
   The protocol consists of three phases:

   (a) Periodic listen and sleep (TDMA)
   (b) Collision and overhearing avoidance
   (c) Message passing

   In S-MAC, each node has its own listen / sleep schedule and these schedules are synchronized among the neighboring nodes. An adaption to S-MAC is Timeout-MAC (T-MAC), which extends S-MAC by making it adaptive to traffic variation.

2. Carrier Sense Medium Access with Collision Avoidance with reservation [30] uses Ready To Send (RTS) and Clear To Send (CTS) messages. The basic protocol functionality is the same as in CSMA. The extension is that if the channel is sensed idle, a node will first broadcast a Ready To Send message. The destination node will transmit a Clear To Send message if the destination node is not currently involved in transmitting data. The sending node will receive this CTS message - if transmitted - and will continue with the transmission, thereby solving the hidden node problem.

#### 2.3.2.3 Class 3 MAC protocol

The basic solution used in class 3 protocols is the usage of RTS and CTS messages, but in a more advanced manner than CSMA with reservation. Two protocols using this technique are the Dual Busy Tone Multiple Access (DBTMA) protocol and the Medium Access Collision Avoidance (MACA) protocol.

1. The Dual Busy Tone Multiple Access protocol [28] is a TDMA-based protocol, that uses two channels to prevent the hidden / exposed node problem from occurring.

   (a) One channel for data packets and Ready To Send (RTS) / Clear To Send (CTS) information
   (b) One channel for busy tones

   When a source A wants to transmit its data, it checks its own busy tone channel if it is free. If it is, it transmits a RTS signal. If receiving node B receives this, the data bus channel is free. Node B transmits a CTS signal and sets the receive busy tone. Node A receives the CTS signal, sets the transmit busy tone and transmits its data. In this way, the hidden terminal problem is solved because the hidden node sees the transmitter's busy tone and the exposed node problem is solved because the exposed node sees that the exposed node's busy tone is not being influenced by the already transmitting node.

2. The Medium Access Collision Avoidance protocol aims to solve the exposed and hidden node problems by introducing RTS and CTS control messages. If a node wants to transmit a message, it will first issue an RTS message to its destination. If the destination node receives the RTS message, it will issue an CTS packet to the sending node. When the sending node receives the CTS message, it will begin transmitting.
   Now, when a nearby node hears an RTS message, it inhibits its own transmission for a while, waiting for a CTS response. If the CTS response is not heard, the nearby node can start transmitting - and thereby omits the exposed node problem. When a nearby node overhears a CTS message, a node will inhibit its own transmission for a sufficient time to allow the corresponding data communication to complete - and thereby omits the hidden node problem.

## 2.4 Background information conclusion

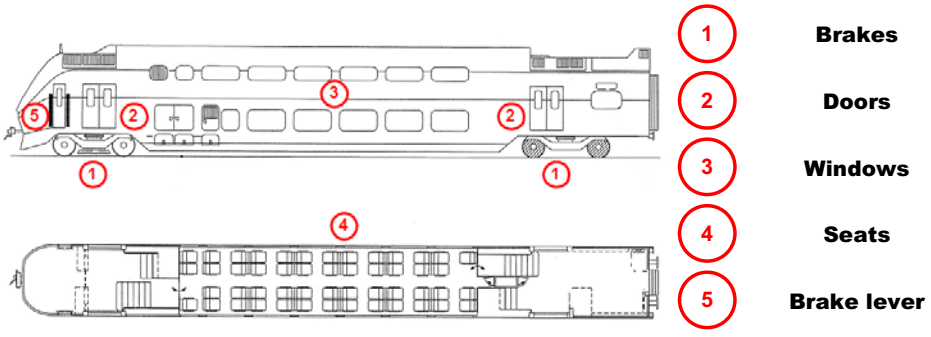
This chapter ends with a discussion on the conclusions that can be drawn from this chapter, which presented the background information necessary to understand this project.
The conclusions from Section 2.1 on hybrid wireless sensor network terminology are that the network components used - the sensor nodes - are small, have limited capacities and can be of different capacity-types (therefor the name 'hybrid'). Because of these limited capacities, a node has a limited bandwidth, a limited amount of energy available and only small computational powers. Besides this, the self-organizing functionality of the nodes offers the nodes ways to connect to each other according to a specific topology and thereby create a sensor network.
Second, the conclusions from Section 2.2 on reliability are that the environment the nodes are deployed in plays an enormous role. Reliability experiments using the most commonly used reliability metrics (being Packet Reception Rate and Received Signal Strength Indicator) showed undeterminable behavior - a phenomenon called 'radio-irregularity'. This means there are not 2 regions in a link between 2 nodes (either connected or not-connected with a PRR of resp. 100 % and 0 %), but 3! This third region, the transitional region, lies between the connected and non-connected region. The transitional region shows an undeterminable PRR, which varies between 100 % and 0 %. This can have several reasons, which have to do with the environment or the hardware used. Fortunately, there are theoretical models available that model and explain the existence of these 3 regions using the PRR and RSSI metrics.
Finally, the conclusions from Section 2.3 on the data link and network layer solutions are that reliability can be improved on the data link layer by trying to solve the hidden and exposed node problem. Therefor, a categorization is made in 3 MAC classes - the protocols that either don't solve this problem (Class 1), only the hidden node problem (Class 2) or both the hidden and exposed node problem (Class 3). On the network layer, the protocol solutions improve reliability by a specific routing method. Therefor, a categorization is made in several routing type classes, that all improve reliability in their own way.

# 3 Requirement analysis

This chapter starts with a thorough analysis of the train applications in Section 3.1. These applications tell a lot about the scope of the sensor network implementation. The applications will be used as a starting point for the requirement analysis, which follows after the application analysis. Requirements can be divided into functional and non-functional requirements. For a clear structure, a start will be made with the non-functional requirements. Non-functional requirements are requirements that specify criteria that will be used to judge the operation of the sensor network. These requirements are described in Section 3.2. The functional requirements define the functionality the sensor network must have. They are described in Section 3.3. This chapter ends with a conclusion on the requirement analysis in Section 3.4.

## 3.1 Train application analysis

This section discusses the applications inside a train the sensor network can be used for, with a focus on condition monitoring and customer service. Each application is briefly discussed by analyzing the background of the application, its usage in a wireless sensor network and the location of the application within a Dutch Railways train, which is shown in Figure 3.1(a) [8]. Finally, the number of data samples that have to be transmitted to update the application information is discussed. The numbers discussed are based on the experience of Logica engineers.




(a) Train ground plan [8] (b) Legend

Figure 3.1: Overview train applications

All applications described use the low-capability, transmit-only sensor nodes.

### 3.1.1 Condition monitoring

Conditions are monitored to prevent material from breaking down by recognizing material wear out. When wear out is indeed recognized in time, it can be repaired without causing problems when the material breaks down when it is functioning. The following applications focus on condition monitoring in trains.

#### 3.1.1.1 Air conditioning control

Trains are being used in every kind of weather, varying almost by the minute. The number of people inside the train changes even more - from absurdly crowded in rush hour to very calm outside these hours. To increase people's comfort, the train is usually equipped with air conditioning. Air conditioning is located near every window. The lay-out of the windows is shown as number 4 in Figure 3.1(a) [8].
The air conditioning measures the temperature and reacts when necessary by pumping dry air in the train.
A wireless sensor network can be implemented here to acquire more data on temperature / humidity of the air from more diverse places in a train compartment. Even though the sensor nodes are on fixed places, this still makes these nodes easier to be placed, since they have no fixed cables or a power supply.
Large temperature changes do not occur that quick, so a sampling time of 1 sample / 2 minutes will be enough to monitor a temperature change and respond in time.

#### 3.1.1.2 Door malfunctioning

Trains are long and have many doors to help people enter or leave the train quickly. The lay-out for doors is shown as number 2 in Figure 3.1(a). They are positioned at the far ends of the train compartments, opposite each other. Trains are not allowed to leave a station when there are still doors left open; obviously, this is not safe for both passengers as well as the train staff.
To help the train staff to find an open door easier a so-called "green light"-system is used to monitor doors. This system consists of a serial circuit between all doors. A green light indicates that all doors are closed. If this light is not on, the train is not allowed to leave the station. Problem here is that the system indicates that there is a malfunctioning door, but it does not indicate where. Thus, if the green light is off, all doors must be verified for correct functionality - manually, by a conductor. Needless to say that this consumes much time and causes delays.
A wireless network can be used to solve this problem by installing sensor nodes near the doors, that indicate if that specific door is functioning correctly. As a result, the train staff will be informed on a mobile device to see which door is malfunctioning.
Door malfunctioning only needs to be checked when the train is halted, but if it is, it must be monitored quickly. A sampling time of 1 sample / 2 seconds will be enough to recognize and monitor a malfunctioning door in time.

#### 3.1.1.3 Brake lever position

Slowing a train down by braking it is initiated by applying a brake lever. This lever is mounted at the head and tail of a train. Number 5 in Figure 3.1(a) shows the location in the train. Figure 3.2 [8] shows this device. Braking is done by manually moving the

Figure 3.2: Brake lever [8]

lever into either the non-overrulable or the overrulable position.

1. In the non-overrulable position, the brake lever cannot be overruled at any other position in the train.

2. In the overrulable position, the brake lever can be overruled at any other position in the train.

If the brake itself is in this non-overrulable mode the other brake cannot be used any more. Practically, this means that it can be unclear which brake is being used and which brake can be used. The situation can occur that the train staff has to walk through the train to find the non-overrulable brake, before the train can start moving again. This will take some time, which will cause the train to be delayed.
A wireless network can be used to solve this problem by installing sensor nodes that monitor the position of the brake. The train staff will receive this information and can respond by removing the brake. Just like in all applications so far, all nodes will have a fixed position.
Again, this problem only needs to be solved when the train is halted and again, it must be recognized quickly - a sampling time of 1 sample / 2 seconds will be sufficient.

#### 3.1.1.4 Brake condition monitoring

Maintenance on trains is done when it is necessary. Predicting this event, telling when this has to be done saves much time and money; instead of not using a train on a busy day the train is repaired when there is time available and the train is not needed. Predicting maintenance on e.g. brakes is done by monitoring vibrations or temperatures. This gives a good indication of the current status of the brakes. Some of these principles are

already implemented on modern trains, but many old trains are still running every day. Equipping these trains with modern technology solves the described problems, even for old trains. Number 1 in Figure 3.1(a) indicates where train brakes are.
The (fixed) nodes of the sensor network can help in this example by measuring the temperatures and vibrations and alerting the workshop they have to take action.
Just like in the air conditioning application, train brakes do not break down that suddenly - there is a slight deterioration and eventually, this deterioration will fall below a certain threshold below which the brakes simply do not function sufficiently any more. In this case, it is time to replace the brakes. When looking at sample times, a sampling time of 30 samples / hour will be enough to monitor the temperature and be able to analyze the data to find out if it is indeed below a critical threshold.

### 3.1.2 Customer information

Information on seat occupation is very useful to inform both passengers and railway companies. Sensor nodes can be placed inside train seats (see number 4 in Figure 3.1(a)) to monitor the occupancy at that very moment. Example applications are traveler service and statistical applications.

1. In traveler service applications, a sign can be used to indicate to passengers if there are free seats left in the compartment. Signs in the train or on the station can be used to guide passengers to a free seat. This helps improve the flow of people through the train. As a result, the time it takes for people to leave or enter the train will decrease.

2. In statistical applications, monitoring how many people are traveling in one period of time, e.g. rush hour gives useful information. Knowing this gives much insight into the logistical problem railway companies have to cope with. Based on this information, decisions can be made when and where more trains are needed.

The data of both applications will be most interesting closely before or shortly after a train leaves the station. Since people tend to enter and leave a train in a very short time, the sample time chosen here to monitor each single application is 1 sample / 2 minutes.

## 3.2 Non-functional requirements

The criteria that will be used to judge the quality of the sensor network are the resource constraints and performance.

### 3.2.1 Resource constraints

The resource constraints are derived from the applications discussed in Section 3.1. These constraints focus on the minimum throughput rate required, the run time of the applications, the type of nodes prescribed and the programming language prescribed. Please note that there is no constraint on the reliability level - remember, one of the goals of this project is to calculate and be able to tell what the current reliability level is - not to make this level as high as possible.

#### 3.2.1.1 Minimum throughput rate

*The sensor network must have a throughput rate that is equal to or higher than the minimum throughput rate.*
All applications have a minimum throughput rate - this rate is the minimum number of data packets, providing information of the source, that is needed to be updated in time that the low-capability nodes produce. If the actual throughput rate is less, this will cause the applications to work (partially) incorrectly and this can have huge impact - for example, if the door malfunctioning application (from Section 3.1.1.2) is not working in time, the whole principle of using a sensor network to save time is gone and eventually, the train staff will go back to the old-fashioned method of checking the doors themselves, which will cost an extra amount of time - but it will always produce the correct outcome. The minimum throughput rate can be calculated by adding all the applications' sample times together. The minimum throughput rate, required by the sensor network, is 63 samples / second - see Table 3.1.

| Application (Section) | Samples / minute |
|---|---|
| Air conditioning control (3.1.1.1) | 1/2 |
| Door malfunctioning (3.1.1.2) | 30 |
| Brake lever position (3.1.1.3) | 30 |
| Brake condition monitoring (3.1.1.4) | 1/2 |
| Traveler service (3.1.2) | 1 |
| Statistical purposes (3.1.2) | 1 |
| All together | 63 |

Table 3.1: Calculating the minimum required throughput rate

#### 3.2.1.2 On time

*The sensor network must monitor all applications all of the time.*
Not all applications need to be monitored all the time - for example, the door malfunctioning application only needs to be running when the train is approaching or at a station. This may contradict with the first statement, but since there *are* applications that need to be monitored all of the time (see Table 3.2) - such as the brake condition monitoring application - the choice is made to monitor all of the applications all of the time. Shutting several applications down at a certain time would allow a system to be integrated that would tell specific nodes not to function. The complexity of this system is far more complex than just ignoring the data of several applications during run time, when they are not needed.

#### 3.2.1.3 Used node types

Both backbone and application nodes use different node types.

1. *The backbone nodes must be high-capability sensor nodes.*

| Application (Section) | On time |
|---|---|
| Air conditioning control (3.1.1.1) | Always |
| Door malfunctioning (3.1.1.2) | Braking |
| Brake lever position (3.1.1.3) | Braking |
| Brake condition monitoring (3.1.1.4) | Always |
| Traveler service (3.1.2) | Always |
| Statistical purposes (3.1.2) | Always |

Table 3.2: Applications on time

2. *The application nodes must be low-capability sensor nodes.*

#### 3.2.1.4 Programming language

*All programs must be written in nesC.*
NesC is the programming language of TinyOS, the OS of the Crossbow Mica2 sensor nodes and therefore, all programs must be written in nesC.

#### 3.2.1.5 Project documentation

*All documentation must be written in English and documented using both Word- and PDF-files.*
This thesis is the complementation of a Masters in Science-program which is completely lectured in English. Besides that, Logica wants to have a copy in a Word-file, whereas the TU Delft wants to have a copy in a PDF-file.

#### 3.2.1.6 Availability

*Functionality that is already available must be used over functionality that must be designed hand-made.*
It might very well be possible that certain protocols have already been designed or can be downloaded from an Internet site somewhere. This is a major advantage since it will save much time - instead of designing a new protocol, an already existing one can be used.

#### 3.2.1.7 Design documentation

*All functionality must be properly documented or already have good documentation.*
All functionality that is either already existing or to be designed must have good documentation. This project serves as a starting point for other projects and readability is a key asset.

### 3.2.2 Performance

Section 2.1.1.1 showed that the node's capacity is limited due to its small size. The small size limits the amount of memory and only allows it to have a small (and thus relatively

slow) microprocessor. As a result of this, the node's bandwidth and throughput are limited.

#### 3.2.2.1 Measurements

At a certain moment in time, the node will start losing data packets, simply because it cannot cope with the amount of work it has to do - receiving data, processing data and transmitting data at the same time. Section 3.2.1.1 stated what the minimum required throughput rate is - this is the data packet rate that is actually received by the receiving node. Throughput is calculated by using the following theorem.

**Theorem 3.1**

$$Throughput = PRR \times transmission\ rate$$

1. *The transmission rate must be monitored by the transmitting node.*
   The transmission rate is the rate, with which the transmitting node transmits data packets.

2. *The PRR must be monitored.*
   The PRR is the rate of the transmitted data packets that have been correctly received by the receiving node.

3. *The actual throughput rate must be monitored.*
   The throughput is the multiplication of the PRR and transmission rate. This rate must be equal or higher than the minimum throughput rate described in Section 3.2.1.1.

#### 3.2.2.2 Use different setups

Eventually, the node backbone configuration will be put under a performance test, where the backbone node configuration uses high-capability nodes and the applications use low-capability nodes.The backbone configuration was already discussed in Section 1.1, but now the applications have their influence as well. To be able to tell the influence of the specific components, several node setups have to be used to reach this goal.

1. *A single link setup must be evaluated.*
   The first step is to find out what the maximum throughput rate is for a single link setup. If first node is only transmitting and the second node is only receiving, this will tell what the maximum transmission and receiving speeds are. If the throughput requirement is not met using a single link, it will surely not be met in the backbone setup.

2. *A multiple link setup, using 2 links, must be evaluated.*
   The second step is to find out what the maximum throughput rate is for 2 links by lining up 3 nodes in a linear array. By doing so, this will tell what the maximum transmission and receiving speeds are of a node that is both transmitting and receiving.

3. *A multiple link setup, using 7 nodes, must be evaluated.*
   Since there are 8 nodes available, the previous experiment can be easily expanded to a 7 link node setup, to find out if extra links increase reliability.

4. *The backbone setup must be evaluated.*
   Finally, the backbone setup must be evaluated with the applications attached: 3 nodes serve as the backbone node configuration, with 4 nodes (serving as the applications) transmitting to the backbone nodes. Results from this experiment will finally tell if the performance requirement is met - or not.

### 3.2.3 Non-functional requirements summary

Table 3.3 summarizes all the non-functional requirements described in this section.

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Resource constraints | None | Throughput rate $\geq$ 63 samples / min. | 3.2.1.1 |
| | | Always monitor all applications | 3.2.1.2 |
| | Used node types | Backbone nodes: high-capability<br>Application nodes: low-capability | 3.2.1.3 |
| | None | All programs in nesC | 3.2.1.4 |
| | | English Word- & PDF-documentation | 3.2.1.5 |
| | | Use available functionality | 3.2.1.6 |
| | | Provide design documentation | 3.2.1.7 |
| Performance | Measurements | Monitor transmission rate<br>Monitor PRR<br>Monitor throughput rate | 3.2.2.1 |
| | Use multiple setups | Use single link<br>Use 2 link<br>Use 7 link<br>Use backbone configuration | 3.2.2.2 |

Table 3.3: Non-functional requirements summary

## 3.3 Functional requirements

This section defines the functional requirements of this projects. The functional requirements are based on reliability and the desired functionality in the layers applicable to this project.

### 3.3.1 Reliability

Again, reliability is measured to be able to tell what the current reliability level of the sensor network is. Remember, focus here is on knowing what the current reliability level is - not on trying to achieve a certain level of reliability. Therefore, the reliability requirements are placed under the functional requirements - all requirements on reliability

focus on knowing what the current reliability level is.
First of all, the used environment is the dynamic train environment. Second of all, a profile must be made of the this environment, which will tell what the reliability of the environment is using a single link. The theoretical section on reliability (in Section 2.2.1) showed that the metrics most commonly used for calculating reliability are the Packet Reception Rate and the Received Signal Strength Indicator.

#### 3.3.1.1 Used environment

*The condition monitoring application, using the high-capability and low-capability nodes, must function correctly inside the dynamic train environment.*
However, due to the dynamic character of this environment, the best way to validate all tests is to perform all tests in a static environment with no interfering or changing obstacles. However, all final tests must be conducted inside the dynamic train environment.

1. *For validation only, all experiments must be conducted inside a static environment.*

2. *All final experiments must take place inside the dynamic train environment.*

#### 3.3.1.2 Single link profile

*The standard environment profile is based on a single link between two nodes.*
The primary setup of this project is a linear array, which serves as a backbone for low-capability node applications inside the train. To find out what the right amount of space between two nodes is, an environment profile will be made of the dynamic train environment.

1. *The environment profile must show where the transitional region starts.*
   Since the transitional region is the region in which data packets are not guaranteed to be delivered (remember, the PRR can vary between 1 and 0 in the transitional region) it is important to know at what distance the transitional region starts. Knowing this will give valuable information on what the maximum distance is that nodes can be separated.

2. *The environment profile must show what the path loss exponent of the standard environment is.*
   If the path loss exponent of the environment is known, predictions can be made on what the maximum distance between two nodes is. This is needed for the transitional region calculations.

#### 3.3.1.3 Metrics used

The two metrics most commonly used are Packet Reception Rate (PRR) and Received Signal Strength Indicator (RSSI).

1. *The standard environment profile calculations must be based on PRR.*
   If the PRR profile is known in a standard environment, then this information can

be used in the next step: a train environment. Without any obstacles, interfering signals etc. the maximum distance nodes can be apart without being in each other's transitional region. Therefore, the PRR must be monitored.

2. *The standard environment profile calculations must be based on RSSI.*
   Besides the PRR, the standard environment profile will also be based on RSSI measurements.

### 3.3.2 Layer requirements

The layer functional requirements are based on the OSI model. The OSI model is a layered, abstract description for communications and computer network protocol design, consisting of seven layers. This model is used since it provides a framework for extracting requirements. For more details, see Section A.4. The two OSI layers applicable to this project are the network layer and the data link layer.
Section 3.2.1.1 showed there is a minimum required throughput rate of 63 samples / minute. To find out if the performance requirements are met, the mechanisms on these two layers will be benchmarked to find out if their current performance level is sufficient - or not - to achieve the required performance values. Besides benchmarking the current implementation, these performance values will also be compared with other mechanisms on the same layer, that perform the same functionality - but in a different way. For example, both static and dynamic routing route data units to a destination node - but they are based on a totally different basic principle.

#### 3.3.2.1 Network layer

Section A.4.1.2 stated the functionality of the network layer:

> *The network layer provides mechanisms for routing the data units across the network.*

Routing provides the mechanisms necessary to deliver a data packet at the correct recipient.

1. *A connection must be established, if the node's application layer indicates so.*
   The routing algorithm must route the data from the transmitting to the receiving node.

2. *The routing mechanism must route the data to the destination node.*

3. *Re-routing must be performed, if the node's upper layers tell so.*

4. *The routing algorithm must provide the quality level of the connection.* This information is needed by the upper layers. The source and destination node must provide the status of the link between them.

   (a) *If data has been correctly received, the node must indicate so.*

   (b) *If data has not been correctly received, the node must indicate so.*

#### 3.3.2.2 Data link layer

Section A.4.1.2 stated the functionality of the data link layer:

> *The data link layer provides mechanisms for accessing the shared physical medium by the device.*

Accessing a shared physical medium is the objective of the Medium Access Control. The Medium Access Control mechanism controls the access of a node to the shared medium and must make sure that only one node at a time controls the shared medium.

1. *No multiple nodes are allowed transmit data at the same moment.*
   Otherwise, packet lose due to collisions will occur.

2. *If the channel is indeed free, the node is allowed to transmit its data.*

3. *If the channel is not free, the node is not allowed to transmit any data.*
   If the channel is not free and the node is allowed to transmit data, a collision will occur.

4. *If the channel is not free, a certain amount of time must pass before a retry can take place.*
   The amount of time depends on the protocol implemented. Chances are very high that the channel is still occupied if it tries to transmit its data again immediately.

### 3.3.3 Functional requirements summary

Table 3.4 summarizes all the functional requirements described in this Section.

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Reliability | Used environment | Validation only: static<br>Final test: dynamic | 3.3.1.1 |
| | Single link profile | Start transitional region<br>Derive path loss exponent | 3.3.1.2 |
| | Metrics | Measure PRR<br>Measure RSSI | 3.3.1.3 |
| Network layer | Routing | Establish connection<br>Route data<br>Re-route if necessary<br>Provide quality level | 3.3.2.1 |
| Data link layer | MAC mechanism | No multiple transmitting nodes<br>Permission if channel is free<br>No permission if channel is not free<br>Wait before retry | 3.3.2.2 |

Table 3.4: Functional requirements summary

## 3.4 Requirement analysis conclusion

This chapter ends with the conclusions that can be drawn from this chapter. This chapter analyzed the applications that will use the backbone network for data transport. These applications were used as a basis for a requirement analysis, which was divided in a non-functional and functional part.
Section 3.1 on the train application analysis showed all nodes have fixed positions. The backbone nodes have a power supply, so there is no need to focus on energy savings. There are 2 types of applications that will use the backbone nodes - condition monitoring and customer information applications. Each application has its own required minimum throughput rate and produces a continuous data stream. Besides this, the application nodes will use the low-capability nodes. The backbone nodes however will be high-capability nodes.
Section 3.2 on non-functional requirements showed that together, all applications require a backbone node minimum throughput rate of 63 samples per minute, where 1 sample means 1 data packet transmitted. Second, a series of setup configurations must be used to find out what the performance level of the backbone node configuration and the influence of the applications is. The non-functional requirements are summarized in Table 3.3.
Section 3.3 presented the functional requirements by focusing on reliability and layer functionality, which are summarized in Table 3.4. It was concluded that even though the final tests will be in a dynamic environment, the best way to start is by performing tests in a static environment to exclude external influences. Besides this, to find out what reliability means, a profile must be made of the used environment that analyzes the transitional region and calculates the path loss exponent by using the PRR and RSSI metrics.

# 4 Solution selection

This chapter discusses how to fulfill the requirements in Chapter 3, by matching them with the alternative solutions from Chapter 2. First of all, Section 4.1 discusses how the reliability requirements are fulfilled. Following this, Section 4.2 discusses which network mechanism(s) best fulfill the network layer requirements and selects the corresponding mechanisms. Third, Section 4.3 does the same for the data link layer. This chapter ends with a conclusion on the solution selection in Section 4.4.

## 4.1 Reliability calculation solutions

This section discusses the reliability calculation solutions, which solve the requirements on reliability calculation. These requirements were described in Section 3.3.1. For convenience, they are summarized in Table 4.1.

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Reliability | Metrics | Measure PRR<br>Measure RSSI | 3.3.1.3 |

Table 4.1: Reliability requirements summary

The single link profile will be presented in Chapter 5, the evaluation chapter. This section describes the proposed methods on how to calculate the Packet Reception Rate and RSSI. Before doing so, let's take a closer look at the TinyOS message structure, which is very helpful in providing a (partial) solution to this problem. Figure 4.1 shows the contents of a TinyOS message structure, which can be found in the file 'AM.h'[1]. This section will now continue with a discussion on how to measure or calculate both the PRR and RSSI.

### 4.1.1 Calculate PRR

The Packet Reception Rate (PRR) is calculated as the ratio between received and sent packets; it is measured as the percentage of packets that arrived undamaged on a link - see Theorem 2.1, which is repeated here.

$$\text{Packet Reception Rate} = \frac{\text{\# packets received by node}}{\text{\# packets sent to node}}$$

Looking at this formula, this involves both the transmitting as well as the receiving node to keep track of specific data needed to calculate the PRR.

[1] For a description of programs and files used, see Appendix A.3.

```
typedef struct TOS_Msg
{
  /* The following fields are transmitted/received on the radio. */
  uint16_t addr;
  uint8_t type;
  uint8_t group;
  uint8_t length;
  int8_t data[TOSH_DATA_LENGTH];
  uint16_t crc;

  /* The following fields are not actually transmitted or received
   * on the radio! They are used for internal accounting only.
   * The reason they are in this structure is that the AM interface
   * requires them to be part of the TOS_Msg that is passed to
   * send/receive operations.
   */
  uint16_t strength;
  uint8_t ack;
  uint16_t time;
  uint8_t sendSecurityMode;
  uint8_t receiveSecurityMode;
} TOS_Msg;
```

Figure 4.1: TinyOS message structure

#### 4.1.1.1 PRR to transmitting node

The transmitting node needs to know how many packets it transmitted to a specific node. Once a packet is transmitted to a certain node, it must include the number of data packets transmitted to that node in the data packet transmitted[2]

#### 4.1.1.2 PRR to receiving node

The receiving node needs to know how many packets it received from a specific node. Once a packet has been received from a certain node, it must raise an internal counter to know that a data packet has been received from that specific node.[3]

### 4.1.2 Calculate RSSI

Retrieving the RSSI value takes a little more work. The TinyOS message structure contains a 'strength'-field, which contains the digitized value of the analogue received

[2]If the counter used to calculate the number of packets is saved, a new problem may arise: the problem of overflow. The 'data'-field is only 8 bits wide and is unsigned; therefore, it has a maximum value of 255. One can imagine that this value is easily reached. If a counter with a larger maximum value is needed, bit shifting must be used - see Section 4.1.2.2 which uses this technique.

[3]The same problem as in Section 4.1.1.1 may rise at the receiving node's side - see Footnote 2.

signal strength. Using this value may seem like a simple task, but there are 3 problems that have to be solved.

1. The first problem is already displayed in Figure 4.1 - the 'strength'-field is among the fields that are not actually transmitted or received on the radio, but only used internally.
2. Second problem is that the 'strength'-field is a 16 bit value - however, this is only an 'internal' value; it is not transmitted. The data-array, however, is transmitted, but this array only contains 8 bit data items.
3. Third problem is that even *if* the 'strength'-field is known, it still remains unclear what it actually means.

#### 4.1.2.1 Internal use

The 'strength'-field is used only internally - therefore, the only solution available is to copy the 'strength'-field into a part of the message that is actually being transmitted, every time a data packet has been received. The 'data'-field is the best place for storing the 'strength'-bytes.

#### 4.1.2.2 16 bit value

Each single item in the data-array is 8 bits wide and the 'strength'-field is 16 bits wide. Solution to this problem is bit shifting - in this way, one 16 bits data field can be stored in two 8 bits data fields. Figure 4.2 shows the source code. Once a data packet has

```
TOS_Msg beacon_packet;

event TOS_MsgPtr ReceiveMsg.receive(TOS_MsgPtr pMsg)
{
  beacon_packet.data[1] = (pMsg->strength)>>8;
  beacon_packet.data[2] = (uint8_t)(pMsg->strength);
}
```

Figure 4.2: Bit shifting the 'strength'-field

been received, its signal strength is saved in the 'data'-field of the 'TOS_Msg' message structure 'beacon_packet'.

#### 4.1.2.3 Use formula

Last problem is that the 'strength'-value is now known - but it remains unclear what it actually means. Fortunately, the sensor node's data sheet [6] provides more detailed information. The analogue RSSI value is sampled by a 10 bit A/D converter, whose digital outcome is saved in the 'strength'-field.
To calculate the node's RSSI value (in volts), the formula from Theorem 4.1 must be used.

**Theorem 4.1**

$$\mathrm{V_{RSSI}} = \mathrm{V_{bat}} \times \frac{\text{RSSI value}}{1024}$$

where:

1. $V_{RSSI}$ is the voltage level of the analogue RSSI signal;
2. $V_{bat}$ is the voltage level of the node's power source;
3. RSSI value is the value in the message structure's 'strength'-field;

This raises a new problem; apparently, the voltage level of the node's power source needs to be known. Normally, this is a battery and the voltage level is decreasing - on the other hand, the node can also have a fixed power source and in this case, the voltage level stays the same. In either case, the node provides the solution itself - it simply uses the A/D converter to measure the current power level.
Now that the voltage level is known, the formula from Theorem 4.2 must be used to calculate the RSSI value.

**Theorem 4.2**

$$\mathrm{RSSI(dBm)} = -51.3 \times \mathrm{V_{RSSI}} - 49.2$$

where:

1. RSSI(dBm) is the Received Signal Strength Indicator-field, in dBm;
2. $V_{RSSI}$ is the Received Signal Strength Indicator-value, in volts - from Theorem 4.1.

## 4.2 Network layer solutions

This section discusses the network layer solutions, that solve the requirements on the network layer. Just to refresh one's memory, the functionality of the network layer is stated here once again:

> *The network layer provides mechanisms for routing data units across the network.*

Section 2.3.1 showed various network layer solutions and this section matches these network layer requirements with these solutions. This section will now continue with an overview of the network layer requirements.

### 4.2.1 Requirement summary

The network layer functional requirements were described in Section 3.3.2.1 and are summarized in Table 4.2. Besides these, there are also a number of non-functional requirements applicable, which were described in Section 3.2.1 and are summarized in Table 4.3. This section will continue with a selection of the best-suited routing mechanism.

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Network layer | Routing | Establish connection<br>Route data<br>Re-route if necessary<br>Provide quality level | 3.3.2.1 |

Table 4.2: Network layer functional requirements summary

| Category | Requirement summary | Section |
|---|---|---|
| Resource constraints | Use available functionality<br>Provide design documentation | 3.2.1 |

Table 4.3: Network layer non-functional requirements summary

### 4.2.2 Routing mechanism selection

Section 2.3.1 showed that various types of routing are available. This section will show which type (and eventually which protocol) will be selected for evaluation in Part 5. All routing types discussed are summarized in Table 4.4, together with their credit points on each requirement. The routing type index is shown in Table 4.5. All routing types

| Index | Routing type | Section |
|---|---|---|
| 1 | Metric based | 2.3.1.1 |
| 2 | Multi-path | 2.3.1.2 |
| 3 | Static | 2.3.1.3 |
| 4 | Geographic | 2.3.1.4 |
| 5 | Source routing | 2.3.1.5 |
| 6 | Moving node routing | 2.3.1.6 |

Table 4.4: Routing mechanism index table

are given the same amount of credits for the first 3 requirements; all 3 items are functionality that just needs to be implemented and are not dependent on the routing type. Fortunately, the TinyOS software provides all this functionality in included libraries, so there should not be any problem here. Each routing type will now be briefly discussed and finally, the best-suited routing mechanism(s) will be selected.

#### 4.2.2.1 Metric based discussion

Metric based routing is, besides geographic routing, the simplest way of routing. The metric defines on which property a next destination node is selected. It's advantages are its lack of complexity - the only problem is of course constructing and storing the metric. Looking at this project, the most obvious metric is of course reliability. However, to use this metric, the node needs to know how many data packets are send to it and how many

| Requirement | | Routing type (index # in Table 4.4) | | | | | |
|---|---|---|---|---|---|---|---|
| Type | Summary | 1 | 2 | 3 | 4 | 5 | 6 |
| Functional | Establish connection | 6 | 6 | 6 | 6 | 6 | 6 |
| | Route data | 6 | 6 | 6 | 6 | 6 | 6 |
| | Re-route if necessary | 6 | 6 | 6 | 6 | 6 | 6 |
| | Provide quality level | 9 | 9 | 6 | 4 | 9 | 4 |
| Non-functional | Use available functionality | 8 | 4 | 8 | 4 | 4 | 4 |
| | Provide design documentation | 6 | 4 | 6 | 4 | 4 | 4 |
| Total score | | 41 | 35 | 38 | 30 | 35 | 30 |

Table 4.5: Routing mechanism selection table

it has correctly received - the number of received packages is easy to calculate, but the number of sent packets has to be included in the sender's data field - as was discussed in Section 2.2.1. If the node has enough data, it can decide who to route its data to. Fortunately, a good reliability-metric based protocol is available: the TinyOS protocol provides a dynamic routing protocol that is based on two metrics: one being the least number of hops, the second being the path reliability. In general, documentation on this protocol is poor, but better than the average documentation available.

#### 4.2.2.2 Multi-path discussion

Using multiple paths seems like a good idea - using two instead of one path will increase the chances of correct reception. On the other hand, this will also increase the amount of complexity; therefore, although this routing type scores good on reliability, the total number of credit points is not the highest.

#### 4.2.2.3 Static routing discussion

Static routing is the simplest way of routing; it involves directly programming the nodes' destination addresses. There is no metric or dynamic algorithm, just static addresses. Advantages are its simplicity. For example, the TinyOS operating system has several libraries available with static routing functionality. Disadvantages are of course that the network is not very flexible - it does not react to changes, therefore it's best in place in either environments where sensor networks perform well in or just for plain testing, to see what the current reliability level is.

#### 4.2.2.4 Geographic routing discussion

Geographic routing is out of scope to this project; there is no need for the backbone nodes to route data to a specific geographic area - they just need to route it to the base node, that is used for communicating the computer that monitors the data.

#### 4.2.2.5 Source routing discussion

Source routing is perfect in environments where nodes frequently stop functioning - for example, whenever batteries wear down; as is the case in the applications attached to the node backbone. Unfortunately, the source routing protocol may not always produce the best path over the least amount of hops.

#### 4.2.2.6 Moving node routing discussion

The application analysis in Section 3.1 showed that besides the backbone, the attached application's nodes have fixed positions as well, and therefore this type of routing is out of scope.

#### 4.2.2.7 Selected routing mechanism and protocol

Table 4.5 shows the grades for all routing techniques, which shows that both static routing and metric-based routing are the best options. These two techniques will be used in the evaluation section in Chapter 5. Interesting here is to find out if the added complexity of a dynamic, metric based routing protocol over a static routing protocol will increase reliability - or not.
Summarizing, one can say that most techniques have good ideas but are simply too complex to make them function correctly - especially in the backbone application, whose sole purpose is to forward all the acquired data. The node application can benefit from complex techniques such as data aggregation, but these techniques are too difficult for the backbone node application.

## 4.3 Data link layer solutions

This section discusses the data link layer solutions, that solve the requirements on the data link layer. For clarity's sake, here is the function of the data link layer.

> *The data link layer provides mechanisms for accessing the shared physical medium by the device.*

Section 2.3.2 showed various data link layer solutions and this section's aim is to match the data link layer requirements with these solutions.

### 4.3.1 Requirement summary

The data link layer functional requirements were described in Section 3.3.2.2 and are summarized in Table 4.6. Besides these, there are also a number of non-functional requirements applicable, which were described in Section 3.2.1 and are summarized in Table 4.7. This section will continue with a discussion selecting the best-suited MAC protocol class.

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Data link layer | MAC mechanism | No multiple transmitting nodes<br>Permission if channel is free<br>No permission if channel is not free<br>Wait before retry | 3.3.2.2 |

Table 4.6: Data link layer functional requirements summary

| Category | Requirement summary | Section |
|---|---|---|
| Resource constraints | Use available functionality<br>Provide design documentation | 3.2.1 |

Table 4.7: Data link layer non-functional requirements summary

### 4.3.2 MAC class selection

Table 4.8 shows an overview of the MAC mechanism requirements, compared with the requirements and the grades obtained for each requirement. For each class, each require-

| Requirement | | MAC class (background section) | | |
|---|---|---|---|---|
| Type | Summary | 1 (2.3.2.1) | 2 (2.3.2.2) | 3 (2.3.2.3) |
| Functional | Prevent multiple transmitting nodes | 3 | 6 | 9 |
| | Permission if channel is free | 6 | 6 | 6 |
| | No permission if channel is not free | 3 | 6 | 9 |
| | Wait before retry | 6 | 6 | 6 |
| Non-functional | Use available functionality | 9 | 6 | 0 |
| | Provide design documentation | 9 | 3 | 0 |
| Total score | | 36 | 33 | 30 |

Table 4.8: Scored MAC mechanism grades

ment will be discussed and a grade will be handed, indicating the quality of coping with the particular problem described in the requirement. Grades range from 0 (extremely poor) to 10 (extremely good). Finally, the best-suited MAC class(es) will be selected.

#### 4.3.2.1 Class 1 discussion

The first class of MAC protocols suffer from hidden and exposed nodes. Example Class 1 protocols are Aloha [18, 19] and CSMA [20].
Class 1 protocols are simple protocols, with little functional overhead. In theory, they should perform well and due to their low complexity, debugging is easier. Both the available functionality as well as the design documentation is good. For example, the Mica2 sensor nodes' standard protocol - that comes with the software - is based on a well-documented, well-available CSMA/CA protocol.

Unfortunately, due to this low complexity, the Class 1 focus is not on reliability and therefor, it can be expected that these protocols' performance is worst of all protocols. For example, the Aloha protocol decides to transmit data whenever it feels like it - if there is a collision, the transmission is delayed for a random period and transmitted again. Obviously, this mechanism cannot prevent multiple nodes from transmitting at the same time.
Aloha is just a simple protocol, but the more complex class 1 protocols - like CSMA do not perform much better. CSMA monitors the active channel for activity, but in a very simple way. And, if the channel is not free, it will wait for a random period, just like Aloha.
Summarizing, the first class does not fulfill the functional requirements well. It does allow multiple nodes to transmit at the same time, and there is no way that a node solemnly can gain channel access - or find out that it cannot gain channel access. There is no way of synchronization between the nodes or an extra channel that is transmitted to alarm the other ones. On the other hand, it gains credit point for its simplicity and availability, which will have to make up for all the downsides the protocol has.

#### 4.3.2.2 Class 2 discussion

The second MAC class solves the hidden node problem, but still suffers from the exposed node problem. An example Class 2 protocol is S-MAC. Although many more Class 2 protocols exist, this discussion focuses on S-MAC. Remarks made here are - more widely seen - valid for all Class 2 protocols.
The S-MAC protocol introduces synchronization - it uses the TDMA-principle to distribute a synchronization schedule among all nodes, so that all nodes know when they are allowed to transmit data - and when not. This also solves the problem of not knowing how long a node must wait before it can transmit data, as long as the node sticks to the synchronized schedule. In general, it can be said that the Class 2 protocols introduce much more complexity than Class 1 protocols. This solves problems - for example, the hidden node problem - but it also introduces complexity problems; for example, debugging the complete protocol is much, much more complex than the Class 1 protocols.
In addition, Class 2 protocol (downloadable) examples are a little hard to find, especially for the TinyOS. There is one available, which is the S-MAC protocol that comes with some example programs and documentation. It is far from the same amount available for the CSMA/CA-protocol in TinyOS (from MAC-class 1), but it can serve as a good start.
Summarizing, the second class partially fulfills the functional requirements, even though it still suffers from the exposed node problem that can cause collisions and data retransmissions. It does score points on the availability and documentation side, but not as much as the Class 1 protocols.

#### 4.3.2.3 Class 3 discussion

The third MAC class solves both the hidden and exposed node problem. All Class 3 protocols described do so by introducing RTS and CTS signals - the DBTMA protocol even uses two channels, with one channel solemnly for busy tones.

Class 3 fulfill the functional requirements best - as the grades in Table 4.8 show, this protocol class would be the best one, if it had not been for the non-functional requirements. Most of these protocols only exist on paper and not in e.g. a downloadable way, so that programmers can use them for testing purposes. This immediately shows the largest disadvantages of these protocols - availability and documentation is low, very low. Knowledge on these protocols is very specific, thus if a company would want to standardize a protocol, it would take a very long time to do so - if this would ever happen. Since there is no source code available, it would take the programmers much time to program the protocol themselves - looking at the context of this project, this knowledge is too specific for the employees of Logica.

#### 4.3.2.4 Selected class and protocol

The grades in Table 4.8 show that the most suitable protocol classes are Classes 1 and 2. It would be very interesting to find out if the added complexity of a Class 2 protocol would be of any use in this project's case - it would have the chance of adding an extra reliability level, but the added complexity would also increase the likelihood of debugging problems. Of both Class 1 and 2, there is source code available for testing - therefore, both the TinyOS CSMA/CA protocol from Class 1 and the S-MAC protocol are selected to be evaluated in Chapter 5.

## 4.4 Solution selection conclusion

This chapter ends with a discussion on the conclusions that can be drawn from this chapter. This chapter discussed how to fulfill the requirements from Chapter 3, by matching the requirements from Chapter 3 with the available background solutions from Chapter 2.
Section 4.1 discussed the reliability calculation solutions, which focus on calculating and deriving the PRR and RSSI. Using these metrics requires little effort. Especially the PRR requires little work - the transmitting node needs to save the number of data packets transmitted, the receiving node needs to save the number of data packets received. Deriving the RSSI requires more work, since it involves both bit shifting and a calculation formula. All together, both of these items should not take much effort.
Section 4.2 discussed the network layer solutions, which focus on selecting the best suitable routing protocol type. Both the static routing and metric-based routing types were selected. The static routing protocol was selected for its simplicity and available (programming library) functionality and documentation. The metric-based routing was selected for its focus on reliability and also its available (programming library) functionality and documentation. The other protocol types were found to be out of scope or too badly-documented.
The data link layer solutions discussion in Section 4.3 selected both Class 1 and Class 2 protocols. Class 1 protocols have many available (programming library) functionality and documentation available. The selected Class 1 protocol is the CSMA/CA protocol. Class 2 protocols have many available (programming library) functionality available as well, but scores a little less well on documentation. However, it does make up in the func-

tional requirements, since it scores much better on reliability than the Class 1 protocol. The selected Class 2 protocol is the S-MAC protocol. The Class 3 protocol was found to be too complex and too badly-documented, even though it scored best on reliability. Therefor, no Class 3 protocol was selected.

# 5 Experimental evaluation

This chapter evaluates the selected reliability solutions and mechanisms from Section 4. First of all, the test environment is discussed in Section 5.1. Section 5.2 evaluates the link quality inside the test environment to find out the maximum distance data can travel reliably. Following this, Section 5.3 evaluates the performance level of the nodes, to find out if the selected mechanisms are actually able to achieve the wanted performance level by the train applications. This chapter ends with a conclusion on the experimental evaluation in Section 5.4.

## 5.1 Experiment environment

This section describes the experiment environment(s) the evaluation takes place in. This section starts with a short summary of the requirements valid for this experiment environment in Section 5.1.1. Following this, the experiment environment(s) are discussed in Section 5.1.2.

### 5.1.1 Requirement summary

This summary shows all experiment environment requirements. All requirements were described in Section 3.3.1 and are summarized for convenience in Table 5.1.

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Reliability | Used environment | Validation only: static<br>Final test: dynamic | 3.3.1.1 |

Table 5.1: Experiment environment requirements summary

### 5.1.2 Used experiment environment(s)

Unfortunately, due to lack of time, the experiments could not be performed inside the dynamic train environment. This subject is left for future research, as will be discussed in Section 6.3. Therefor, all experiments are conducted in a static environment, which is one of the Logica buildings, called 'Beuk'[1]. For a figure of the Logica buildings, see Figure 5.1.

1. All experiments take place in a $12 \times 5$ ($length \times width$, in meters) office room.

[1] During the writing of this document, Logica changed its name from LogicaCMG to Logica and moved from the Kralingseweg in Rotterdam to the George Hintzenweg in Rotterdam. All experiments, however, were conducted in the building located at the Kralingseweg.

Figure 5.1: The Logica building, located in Rotterdam

2. All computer equipment and cellular phones were switched off.

3. No intervening objects are placed within the office unit area.

## 5.2 Link quality

This section evaluates the link quality level of the standard environment used for testing, to find out what the current reliability level is. This section starts with a short summary of the requirements valid for this link quality evaluation in Section 5.2.1. Following this, the experiment outcome will be evaluated and used to fulfill the requirements in Section 5.2.2.

### 5.2.1 Requirement summary

This summary shows all reliability requirements. All requirements are summarized for convenience in Table 5.2. The first two requirements, concerning the used node types,

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Resource constraints | Used node types | Backbone nodes: high-capability<br>Application nodes: low-capability | 3.2.1.3 |
| Reliability | Single link profile | Start transitional region<br>Derive path loss exponent | 3.3.1.2 |
| | Metrics | Measure PRR<br>Measure RSSI | 3.3.1.3 |

Table 5.2: Link quality evaluation requirements summary

are easily fulfilled. Section 1.1 stated that the nodes-to-be used are the Crossbow Mica2 nodes. These nodes are high-capability nodes, but they will be used for the application

type nodes as well.
For convenience, the other requirements are repeated here. These requirements will be addressed in this section.

1. The start of the transitional region indicates the distance at which a node is not able to receive data at the same PRR rate. Remember, there are 3 regions in the transmission range of a sensor node: a connected, non-connected and transitional region. In the transitional region, it is not clear at which rate the node is receiving data correctly - one time, it will have a PRR of 100 %, the other time, it won't receive anything.

2. The path loss exponent $\eta$ indicates the decrease of power of an electromagnetic wave as it propagates through space. Normally, $\eta$ lies between 2 to 4. In some environments, such as buildings and other indoor environments, the path loss exponent can reach values in the range of 4 to 6.

3. The Packet Reception Rate (PRR) is calculated as the ratio between received and sent packets; it is measured as the percent of packets that arrived undamaged on a link. The corresponding formula was stated in Theorem 2.1:

$$\text{Packet Reception Rate } = \frac{\text{\# packets received by node}}{\text{\# packets sent to node}}$$

4. The Received Signal Strength Indicator (RSSI) is a measurement of the power in a received radio signal. The Received Signal Strength Indicator (RSSI) is calculated as the output power of the node minus path loss, as was stated in Theorem 2.2:

$$RSSI(d) = P_t - PL(d)$$

where

   (a) $RSSI$: the signal strength of the received radio signal at distance $d$;
   (b) $P_t$: the node's transmitting power in dB;
   (c) $PL(d)$: the path loss in dB at distance $d$;

### 5.2.2 Setup evaluation

This section validates the outcome of the experiment. First of all, the experiment setup will be discussed. Then, the experiment results will be shown to help verify the theoretical models described in Section 2.2.5.1 to find the path loss exponent and the start of the transitional region.
This section ends with a short discussion on the conclusions that can be drawn from this experiment.

#### 5.2.2.1 Experiment setup

The experiment setup is the following.

1. The experiment took place in the environment described in Section 5.1.
2. The setup uses two nodes: a transmitting and a receiving node.
3. The transmitting node's transmission power is -20 dBm.
4. The position of the transmitting node is fixed.
5. The initial position of the receiving node is 0.5 meter.
6. The distance of the receiving node is increased by 0.5 meter until no data packets are received any more.
7. For every distance the RSSI is measured.
8. For every distance the PRR is measured.

#### 5.2.2.2 Experiment results

Figure 5.2 shows the outcome of the first experiment - the relationship between PRR and distance. The x-axis shows the distance in meters and the y-axis the corresponding PRR. Clearly, up to around 3 meters, the receiving node receives all data packets perfectly - after 3 meters, up to 5 meters, the results are in the transitional region. After 5 meters, there is no reception at all left.

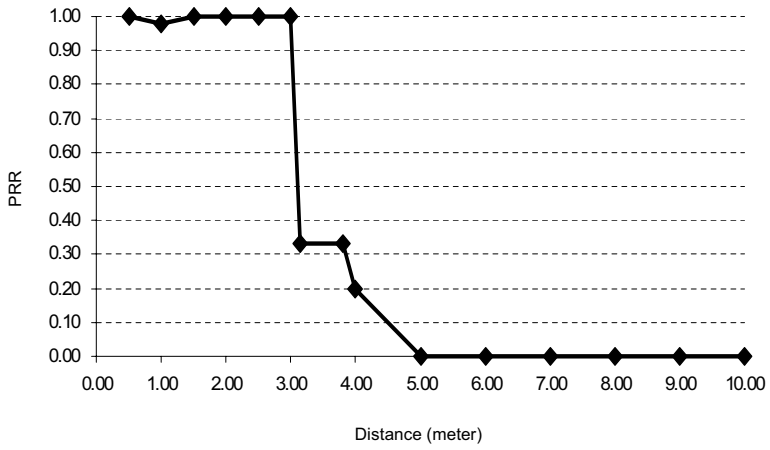



Figure 5.2: Relationship between PRR and distance - measurement values

Figure 5.3 shows the outcome of the second experiment - the relationship between RSSI and distance. The x-axis shows the distance in meters and the y-axis the corresponding RSSI. The figure shows that the decrease of RSSI value quickly decreases from an initial -56 dBm to -95 dBm after 5 meters. After 5 meters, there is no reception possible, due to the high noise level. With RSSI figures larger than -95 dBm, the node is not able to

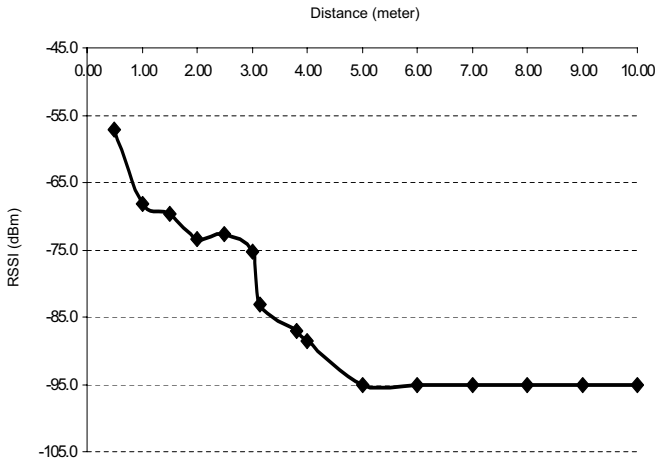



Figure 5.3: Relationship between RSSI and distance - measurement values

correctly receive any data.
These two figures will be used to find the start of the transitional region and the path loss exponent. Three steps are needed to calculate these two items.
The first step is to find the path loss exponent. The path loss exponent is needed to calculate the start of the transitional region and therefore, it has to be calculated first.
The second step is to correct the noise figure model. The noise figure will turn out to be a bit optimistic and therefore it needs to be corrected, since it needs to be used in the transitional regions calculations.
The third and final step is to calculate the start of the transitional region.

1. The first step is to calculate the path loss exponent $\eta$.
   Theorem 2.2 shows the theoretical relationship between the RSSI, the path loss model and the node's transmission power:

   $$RSSI = P_t - PL(d)$$

   where

   (a) The RSSI values are shown on the y-axis in Figure 5.3;
   (b) The transmission power $P_t$ is 20 dBm;
   (c) The path loss $PL(d)$ of the environment can now be calculated.

   The theoretical model of the path loss model is given in Theorem 2.2.5.1:

   $$PL(d) = PL(d_0) + 10\eta \log_{10}(\frac{d}{d_0}) + \sigma$$

   where

   (a) $PL(d)$: the path loss in decibel (dB) at distance d;

(b) $PL(d_0)$: the reference path loss at distance $d_0$;

(c) $\eta$: path loss exponent for the environment that is measured;

(d) $\sigma$: a zero-mean Gaussian Random Variable with standard deviation $\sigma$ in dB - for now, fixed to 0. $\sigma$ will be derived later.

Two graph points are selected in Figure 5.3 to calculate the path loss exponent; this gives the following expressions in Table 5.3. Solving these two equations gives

| RSSI (dBm) | $= P_t$ (dbM) | - PL(d) (dB) |
|---|---|---|
| -69,7 | = -20 | - $(PL(d_0) + 10\eta \log_{10}(\frac{1.5}{1}))$ |
| -95 | = -20 | - $(PL(d_0) + 10\eta \log_{10}(\frac{6}{1}))$ |

Table 5.3: Deriving path loss exponent $\eta$ and reference path loss

the values for the path loss exponent $\eta$ and the path loss value, as is shown in Table 5.4. In general, the calculated path loss exponent shows a good value, since in office spaces the exponent varies between 4 and 6 (see Section 2.2.5.1). Figure 5.4 shows

| Item | Value |
|---|---|
| Path loss exponent $\eta$ | 4,24 |
| Reference path loss | 42,026 |

Table 5.4: Calculated values for path loss exponent $\eta$ and reference path loss

the analytical values compared to the calculated model values. The figure shows something else - a fitting value for the Gaussian variable $\sigma$. Clearly, the calculated model is smoother than the measurement, but they are clearly related. A value of $\sigma = 4$ exactly fits the graph; from now on, this value will be used.

2. The second step is to correct the noise level figure.
   According to the theoretical model in Section 2.2.5.3, the noise model figure the noise level must be near -119.5 dBm in an environment with the conditions as used here. This is the lowest signal level a data packet must have; below this value, the node is not able to process it. But, as Figure 5.3 shows, there is a lot more noise: apparently, the noise level is -95 dBm. From now on, in calculations, for the noise level $P_n$ the value -95 dBm will be used.

3. The final step is to calculate the transitional region.
   The start - and end - of the transitional region can be calculated, using Theorems 2.10 and 2.11. First, the SNR of the start and ending point of the transitional region must be calculated. Theorem 2.9 shows the relationship between a SNR and its corresponding PRR:

$$\gamma = 10 \log_{10}(-1.28 \ln(2(1 - PRR^{\frac{1}{16f}})))$$

where

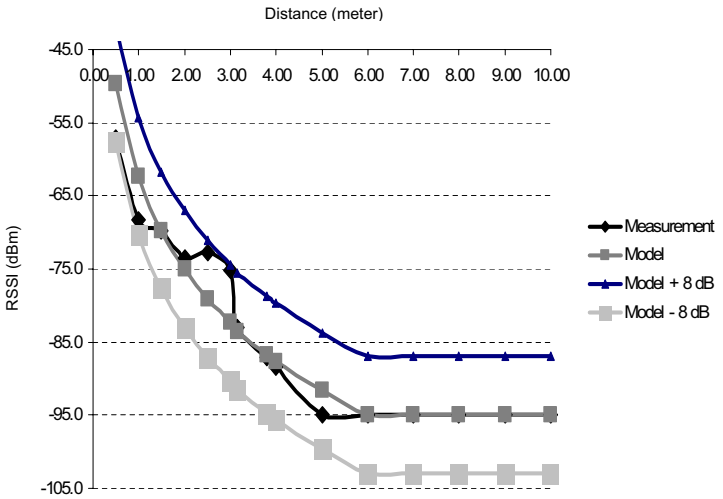



Figure 5.4: Relationship between RSSI and distance - measured and model values

(a) $\gamma$: the SNR rate for the corresponding PRR;

(b) $PRR$: the Packet Reception Rate; here, the values used for correct and incorrect reception must be defined; in most cases, the upper bound value is PRR = 0.9 and the lower bound value is PRR = 0.1.

(c) $f$: the frame size in bytes - remember, a frame consists of a preamble, the data packet itself and the Cyclic Redundancy Check (CRC);

Using this theorem, the following SNR values are calculated - see Table 5.5. Theorems 2.10 and 2.11 calculate the starting point $d_s$ and ending point $d_e$ of

| Item | Value (dB) |
|---|---|
| Upper bound SNR for PRR = 0.9 | 9.9 |
| Lower bound SNR for PRR = 0.1 | 7.6 |

Table 5.5: Calculated values for upper and lower bound SNR values

the transitional region - for the upper and lower bounds of the PRR regions the values 0.9 and 0.1 are selected. The starting point $d_s$ is the following:

$$d_s = 10^{(\frac{P_n+\gamma_u-P_t+PL(d_0)+2\sigma}{-10\eta})}$$

The ending point $d_e$ is the following:

$$d_e = 10^{(\frac{P_n+\gamma_l-P_t+PL(d_0)-2\sigma}{-10\eta})}$$

where

(a) $P_n$: the noise floor level in dB - this value was redefined in Step 2 to -95 dBm;

(b) $\gamma_{0.9}$: the corresponding SNR rate for PRR = 0.9 - see Theorem 2.9 -, which is equal to 9.9 dB;

(c) $\gamma_{0.1}$: the corresponding SNR rate for PRR = 0.1 - see Theorem 2.9 -, which is equal to 7.6 dB;

(d) $P_t$: the node's transmitting power in dB - equal to -20 dBm;

(e) $PL(d_0)$: the reference path loss at distance $d_0$ - calculated in Step 1, equal to 42.026 dB;

(f) $\sigma$: Standard deviation value - derived in Step 1, equal to 4 dB;

(g) $\eta$: path loss exponent for the environment that is measured - calculated in Step 1, equal to 4.24;

Solving these equations gives the values for the start and ending point of the transitional region, which are shown in Table 5.6. Plotting these values on Figure

| Item | Distance (meter) |
|---|---|
| Starting point | 2.25 |
| Ending point | 6.14 |

Table 5.6: Calculated values for transitional region start and ending point

5.2 shows that the values found are correct, possibly even a bit conservative.

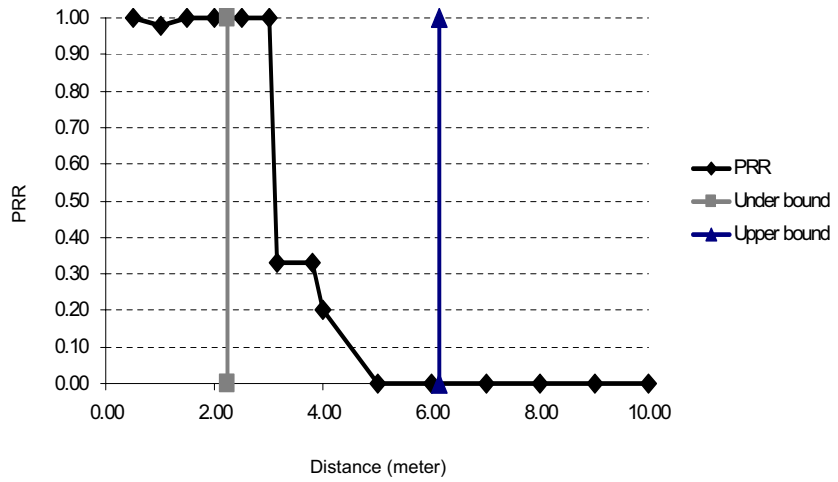



Figure 5.5: Relationship between PRR and distance - measured and model values

#### 5.2.2.3 Experiment conclusion

This experiment has shown that in a static environment, with no interfering obstacles, nodes perform according to the theoretical models described in Section 2.2.5.1. A PRR

and RSSI profile, together with theoretical models, showed that the path loss exponent inside the Logica building is equal to 4. Besides this, using a transmission power of -20 dBm, the transitional region inside the Logica building starts at 2.25 meters. Clearly, this is the maximum distance nodes are allowed to be separated. Of course, the transmission power of the nodes can be increased - this will produce a different PRR and RSSI profile. The method used to calculate the path loss exponent and transitional region, however, stays the same - thus, by performing these two experiments, Logica can gain insight into the performance of nodes inside every environment. Naturally, the static environment here is free of intervening obstacles - this has shown that the theoretical models fit the experiments' outcome. Since these models have proved to be correct, they can now be used inside a dynamic environment, such as the train application.

## 5.3 Performance evaluation

This section evaluates the performance level of the backbone node configuration, to find out if the backbone application can meet the performance requirements. First step is to shortly discuss the mechanisms that will be evaluated in Section 5.3.1. The next step is to show the requirement summary, which summarizes all requirements relevant to the performance evaluation and which will define the setups used in Section 5.3.2. Following this, the nodes' performance level will be evaluated in Section 5.3.3, while using the setups in the requirement section, to recreate the backbone application.

### 5.3.1 Evaluated mechanisms

Table 5.7 shows the mechanisms that will be evaluated in this section. For convenience,

| Category | Mechanism | Section |
|---|---|---|
| Reliability | PRR | 2.2.1 |
| Network layer | Static routing | 2.3.1.3 |
| | Metric based routing | 2.3.1.1 |
| Data link layer | CSMA/CA | 2.3.2.2 |
| | S-MAC | |

Table 5.7: Performance evaluation mechanisms

all items are shortly discussed here.

1. The Packet Reception Rate (PRR) is calculated as the ratio between received and sent packets; it is measured as the percent of packets that arrived undamaged on a link. The corresponding formula was stated in Theorem 2.1:

$$\text{Packet Reception Rate } = \frac{\#\text{ packets received by node}}{\#\text{ packets sent to node}}$$

2. In static routing, nodes are statically programmed to forward data to the next node.

3. Metric based routing decides which node to send its data to based on a metric. This performance evaluation uses the highest reliability / lowest hop count metric.

4. CSMA/CA is a MAC layer, in which each node listens to the channel to see if it's free. If the channel is free, the node will start its transmission. Collisions can still occur. If the channel is not free, it will wait for a random amount of time before it will try to access the channel.

5. S-MAC is a MAC layer, which is a combination of a contention-based and TDMA protocol. Each node has its own time slot. Downside here is that the slot assigning has to be synchronized among the available nodes. On the other had, the protocol saves energy and increases reliability by avoiding idle listening and avoiding hidden node collisions.

### 5.3.2 Requirement summary

This requirement summary shows all performance requirements. All requirements are summarized for convenience in Table 5.8. Throughput will be evaluated using the following setups:

| Category | Sub-category | Requirement summary | Section |
|---|---|---|---|
| Resource constraints | None | Throughput rate $\geq$ 63 samples / min. | 3.2.1.1 |
| | | Always monitor all applications | 3.2.1.2 |
| | Used node types | Backbone nodes: high-capability<br>Application nodes: low-capability | 3.2.1.3 |
| Performance | Measurements | Monitor transmission rate<br>Monitor PRR<br>Monitor throughput rate | 3.2.2.1 |
| | Use multiple setups | Use single link<br>Use 2 link<br>Use 7 link<br>Use backbone configuration | 3.2.2.2 |

Table 5.8: Performance evaluation requirements summary

1. The first setup consists of a single link configuration.

2. The second setup consists of a linear array with 2 links.

3. The third setup will use a linear array with 7 links.

4. The fourth setup consists of the backbone application, using static routing.

5. The fifth setup consists of the backbone application, using metric based routing.

These are the 4 defined setups in Table 5.8, with the addition of a fifth setup concerning the backbone configuration. Due to its high complexity, this setup is used to compare

both static and metric-based routing.
The first 4 setups will be used to find out what the influence of the MAC-layer is. The last setup will be used to compare geographic based routing with metric based routing. Table 5.9 shows an overview of the setups evaluated.

| Setup | Mechanism |
|---|---|
| Single link<br>Linear array, 3 links<br>Linear array, 7 links<br>Backbone, static routing | MAC-layers, using geographic routing |
| Backbone, metric based routing | CSMA/CA, geographic / metric (PRR) based routing |

Table 5.9: Performance evaluation used setups

### 5.3.3 Setup evaluation

The used node types requirements are easily fulfilled. Section 1.1 stated that the nodes-to-be used are the Crossbow Mica2 nodes. These nodes are high-capability nodes, but they will be used for the application type nodes as well.
The other requirements will be addressed in this section.
This section will continue with a discussion of each setup discussed in Table 5.9.

#### 5.3.3.1 Single link configuration

The primary goals of this experiment is to find out what the reliability level of one single link is, using only 2 nodes - without interfering other nodes. Please keep in mind, that these tests only handle with transmitting and receiving data packets - especially in S-MAC, large numbers overhead packets are introduced when synchronization is necessary. This will decrease performance.
This section will continue with the experiment setup, results and conclusions.

1. The experiment is set up in the following way. Figure 5.6 shows the experiment setup, with a circle of radius $r = 2.25$ meter representing the transmission range of node 2.

   (a) All nodes are Crossbow Mica2 nodes, equipped with a CC1000 transceiver.

   (b) In this test, the 'SMACTest' program[2]. is used.

   (c) Two nodes are used: a transmitting node (node 1 in Figure 5.6) and a receiving node (node 2 in Figure 5.6).

   (d) The inner distance between the nodes is approximately 1 meter.

   (e) The *receiving node*'s only occupancy is to be ready and wait for incoming data packets.

[2]For a description of programs and files used, see Appendix A.3.

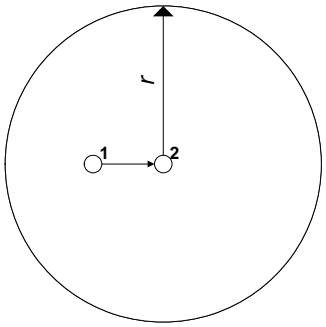



Figure 5.6: Single link experiment setup

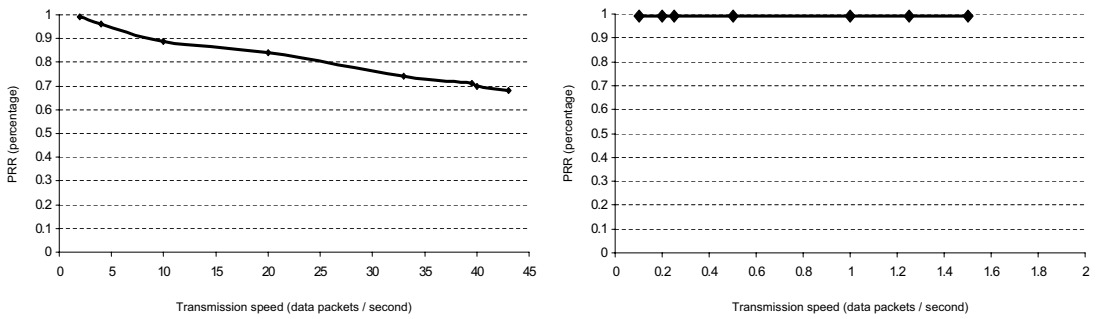
(f) The *transmitting node*'s only occupancy is to transmit data packets at a predefined data rate *trans_rate*. The transmitting node's data packet contains a sequence number *seq_no*, which is raised every time a data packet is transmitted. This sequence number *seq_no* is included in the data packet.

(g) By doing so, the receiving node is able to calculate the number of missing data packets, by simply comparing the number of data packets received internally *count_recv* to the sequence number *seq_no* in the data packet last received.

(h) For each transmission speed, the experiment is performed 10 times and the results are averaged. After this, the transmission speed is raised and the whole sequence is repeated.

(i) The environment itself is the same controlled environment described in Section 5.1, which consisted of an empty office space unit with no intervening objects.

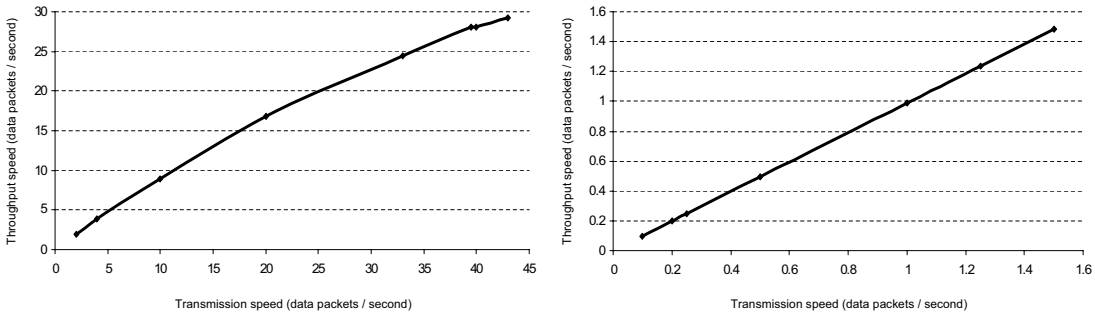
2. The experiment results are the following. For a graph of the test results, see Figure 5.7 for the transmission test results. Figure 5.8 shows the throughput test results.




(a) CSMA/CA results

(b) S-MAC results

Figure 5.7: Single link transmission test results




(a) CSMA/CA results (b) S-MAC results

Figure 5.8: Single link throughput test results

Figure 5.7(a) shows the CSMA/CA transmission test results for the setup shown in Figure 5.6. The x-axis shows the actual transmission speed of node 1 and the y-axis the PRR of node 2.
Figure 5.8(a) shows the CSMA/CA throughput test results for the setup shown in Figure 5.6. The x-axis shows the actual transmission speed of node 1 and the y-axis the throughput speed, measured at node 2.
This figure shows that the maximum transmission rate that can be reached by node 1 is near 43 packets / second. The node will simply not transmit data packets any faster, even if it is programmed to do so. Downside of this relatively high rate is that at this rate more than 30 percent of the transmitted data packets is lost. Most likely, the receiving node is too busy receiving and processing the data that it cannot cope with all data packets. Still, near 70 percent (actually: 69 percent) is correctly received. Thus, as Figure 5.8(a) shows, this gives a maximum reception rate of $45 \times 0.69 = 29.67$ data packets per second.
Figure 5.7(b) shows the S-MAC transmission test results for the setup shown in Figure 5.6. The x-axis shows the actual transmission speed of node 1 and the y-axis the PRR of node 2.
Figure 5.8(b) shows the S-MAC throughput test results for the setup shown in Figure 5.6. The x-axis shows the actual transmission speed of node 1 and the y-axis the throughput speed, measured at node 2.
The one thing that immediately strikes on this figure is the low transmission rate - as Figure 5.8(b) shows, the maximum throughput rate that can be achieved is 1.5 data packet / second = 90 data packets per second - the node will not transmit data any faster. This is little more than the required rate of 63 data packets per minute, which was required (see Table 5.8). Most likely due to its complexity of tranmsitting both data, control and synchronization packets Advantage of the S-MAC implementation is that little to no data packets are lost.

3. The conclusions that can be drawn from this experiment are the following. In this setup, with no other nodes nearby, all data packets transmitted are to be received

by the receiving node. Thus, no effort is wasted on processing data packets that are not intended for the receiving node - as can be seen in Figure 5.6. In this scenario, a maximum throughput rate of 43 data packets (near 30 effectively) is reached by the CSMA/CA-layer. The S-MAC layer can reach up to 1.5 data packet per second and shows almost no losses. Clearly, if all data is to be received, the S-MAC layer is the better choice, but the question is if this layer can cope with the performance requirements in the next experiment scenarios. The large difference clearly lies in the complexity of the S-MAC layer, which - besides the data packets themselves - also regularly transmits control packets and synchronization packets.

#### 5.3.3.2 Linear array using 2 links

The first test showed what the 'pure' transmitting and receiving performance of the sensor nodes was. In the meanwhile, the nodes were not busy processing any data. In this second test, the nodes will - this test's goal is to find out if and in what way this will influence the nodes' performance. This experiment will use a total number of 3 nodes.

1. The experiment is set up in the following way. Figure 5.9 shows the experiment setup, with a circle of radius $r = 2.25$ meter representing the transmission range of node 3.

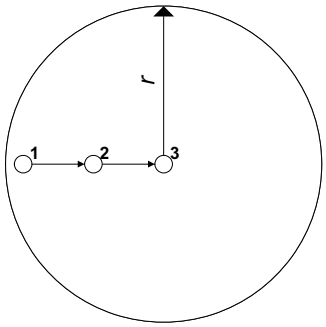



Figure 5.9: 2 links experiment setup

   (a) All nodes are Crossbow Mica2 nodes, equipped with a CC1000 transceiver.

   (b) For testing the S-MAC implementation, the 'SMACTest' program (see Footnote 2) is used.

   (c) All nodes are placed in a linear topology, as can be seen in Figure 5.9. Inner node space is 1 meter.

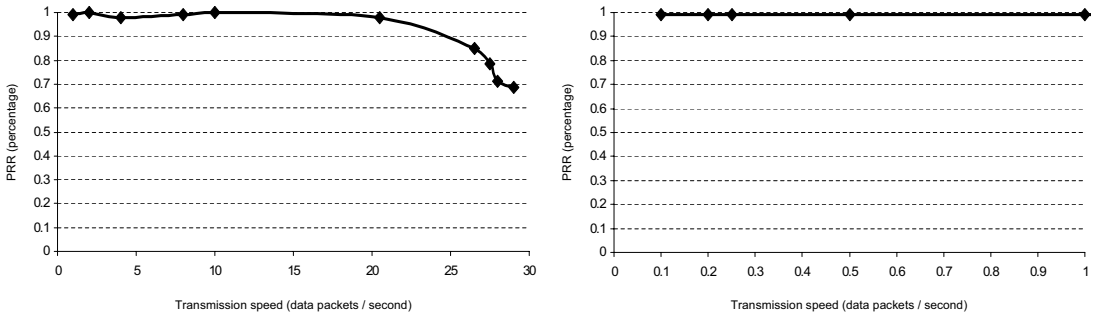
   (d) Node number 1 is used to transmit data to node number 2. The data of the first node consisted of a simple data packet which included a sequence number for verifying the correct order or missing of data.
   Node 2 is the node 'in between'; on one hand, it receives the data from node number 1, which it had to receive. Following this, it would forward node 1's

data package by copying the data and transmitting the data to node number 3.

(e) The environment itself is the same controlled environment described in Section 5.1, which consisted of an empty office space unit with no intervening objects.

(f) The node's data rate is initially fixed to 1 packet per second by programming the internal timer of the sensor node. Due to the changed data stack of the S-MAC protocol, the timing structure has changed as well. The original test program 'SMACtest' which is used for evaluating the node's performance comes with a predefined variable *TST_MSG_INTERVAL* in a configuration file *config.h*, which is used to define the data rate in milliseconds. Unfortunately, the actual data rate value does not turn out to be very precise, due to the face that in the SMAC-implementation not only data but also control and synchronization packets have to be transmitted. It remains unclear which rate is exactly defined by programming *TST_MSG_INTERVAL*.

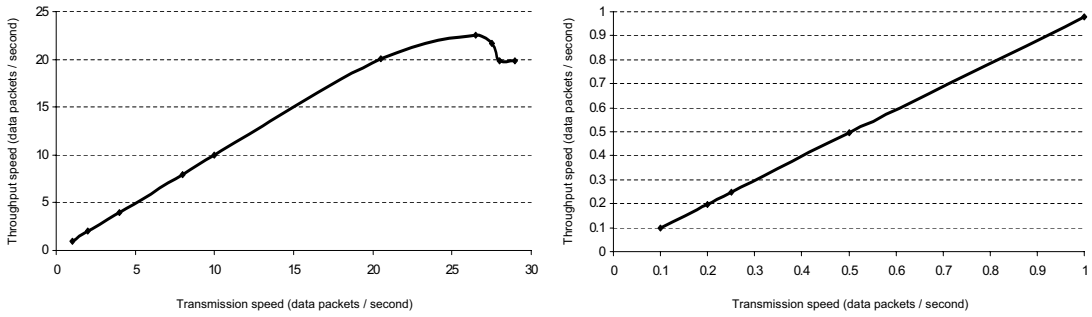
2. The experiment results are the following. For a graph of the test results, see Figure 5.10 for the transmission test results. Figure 5.11 shows the throughput test results, which is the transmission rate times the corresponding PRR.




(a) CSMA/CA results (b) S-MAC results

Figure 5.10: 2 links transmission test results

Figure 5.10(a) shows the CSMA/CA transmission test results for the setup shown in Figure 5.9. The x-axis shows the actual transmission speed of node 1 and the y-axis the PRR of node 3.
Figure 5.11(a) shows the CSMA/CA throughput test results for the setup shown in Figure 5.9. The x-axis shows the actual transmission speed of node 1 and the y-axis the throughput speed, measured at node 3.
In this scenario, node 2 clearly suffers from both having to receive, process and transmit data. This clearly shows in the maximum throughput rate: this rate has dropped to 28 data packets per second. Besides this, the deterioration in PRR is slightly larger than in the single link experiment from Section 5.3.3.1. As Figure 5.11(a) shows, there is a maximum throughput rate of $26.5 \times 0.85 =$




(a) CSMA/CA results (b) S-MAC results

Figure 5.11: 2 links throughput test results

22.5 data packets per second - the node is simply not able to transmit any faster. Remarkably, the maximum throughput rate is not achieved at the maximum transmission rate - clearly, at the maximum transmission rate, the node is too busy processing data to achieve maximum performance. Besides this, the levels of PRR can be compared relatively well with the single link configuration - again, at the maximum throughput rate, the PRR rate is 0.69.
Figure 5.10(b) shows the S-MAC transmission test results for the setup shown in Figure 5.9. The x-axis shows the actual transmission speed of node 1 and the y-axis the PRR of node 3.
Figure 5.11(b) shows the S-MAC throughput test results for the setup shown in Figure 5.9. The x-axis shows the actual transmission speed of node 1 and the y-axis the throughput speed, measured at node 3.
Just as the transmission speed of the CSMA/CA layer is suffering from intervening nodes, so is the S-MAC layer. The maximum transmission rate decreases from 1.5 data packet / second to 1 data packet / second. Still, the reliability level remains at 1 or at 0.99. As Figure 5.11(b) shows, the maximum throughput decreases to 1 $\times$ 0.99 = 0.99.

3. The conclusions that can be drawn from this experiment are the following. Even though the S-MAC layer manages to keep the PRR level at levels very close to 1, it cannot provide the necessary throughput rate needed for the applications in the backbone node configuration. Remember, this is still a simple setup, so performance will even be lower in the backbone scenario. On the other hand, the performance level of the CSMA/CA layer has decreased as well - down to an effective throughput rate of 22.5 data packets per second. This is almost 25 percent less than the results from the single link experiment, but still enough - it is still far more than the required 63 samples per minute (as were required). Once again, the large difference lies in the complexity of the S-MAC layer, which - besides the data packets themselves - also regularly transmits control packets and synchronization packets. Besides this, the setup itself is to blame as well - in the

single link experiment, only 1 node was transmitting - now, there are two nodes transmitting. The link quality experiments conducted in Section 5.2 showed that the maximum distance nodes can be apart with the receiving node being able to correctly receive data was calculated to 2.25 meter. Node 3 is only 2 meters apart from node 1, so node suffers from both node 1 and node 2, as can be seen in Figure 5.9. Node 3 has to process both node's data packets and clearly, this will take up some time. Therefore, the maximum performance rate is less than in the previous experiment.

#### 5.3.3.3 Linear array using 7 links

This experiment's goal is the same as the experiment described in Section 5.3.3.2 - the only difference is that in this experiment, a total number of 8 nodes will be used to further investigate the influence of a large number of nodes both transmitting and / or receiving.
This section will continue with the experiment setup, results and conclusions.

1. Again, this experiment's setup is basically the same as the experiment from Section 5.3.3.2. Figure 5.12 shows the experiment setup, with a circle of radius $r$ = 2.25 meter representing the transmission range of node 8. Since the S-MAC

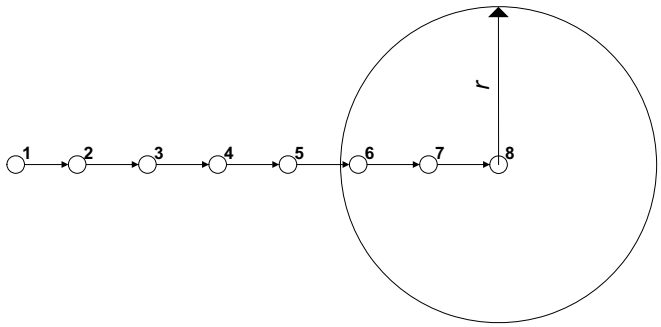
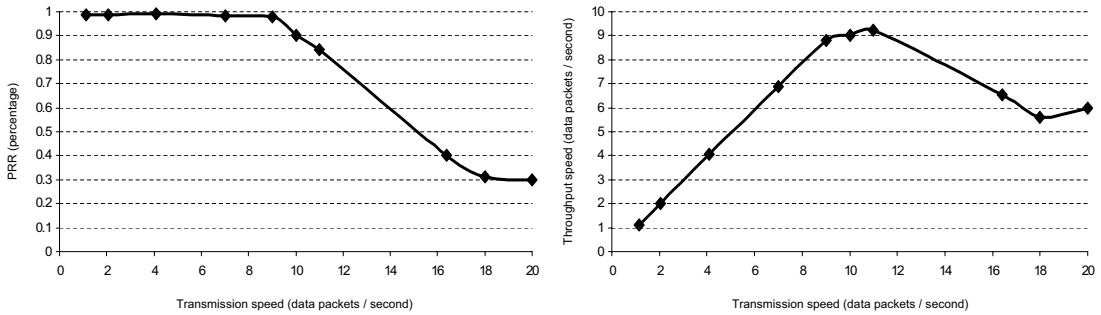



Figure 5.12: 7 links experiment setup

configuration could not fulfill the throughput requirement, this section continues now with only the CSMA/CA layer. Besides this, the only difference is that in this configuration nodes 2 to 7 are the nodes 'in between' and are only used for forwarding data.

2. The experiment results are the following. For a graph, see Figure 5.13.

   Figure 5.13(a) shows the transmission test results for the setup shown in Figure 5.12. The x-axis shows the actual transmission speed of node 1 and the y-axis the PRR of node 8.
   Figure 5.13(b) shows the throughput test results for the setup shown in Figure 5.12. The x-axis shows the actual transmission speed of node 1 and the y-axis the




(a) Transmission test results (b) Throughput test results

Figure 5.13: Test results in a 7 link setup

actual throughput speed, measured at node 8.
Figure 5.13(a) differs from the previous experiments in Section 5.3.3.1 and Section 5.3.3.2 in the way the PRR decreases; previously, there was a similar (single link setup) or nearly-similar (2 link setup) decrease, but in this configuration the PRR remains constant up to near 9 data packets per second. Below this level, the PRR starts decreasing linearly again, to a rather low value.
The performance level is far less than previous experiments; when the nodes are performing at full force (20 data packets), the packet loss rate is near 70 percent! The maximum throughput rate decreases here to $11 \times 0.84 =$ only 9.24 packets per second, as Figure 5.13(b) shows. Once again, the maximum throughput rate is not achieved at the maximum transmission rate.

3. The conclusions that can be drawn from this experiment are the following. Clearly, faster is not always better. Fortunately, this configuration shows that in a linear array, the nodes running on CSMA/CA should have no problem with throughput rates of up to 9 packets per second, which is far more than the required rate of 63 / 60 = 1.05 data packets per second. Once again, the experiment setup is to blame for the performance deterioration. The maximum distance between nodes was calculated in Section 5.2 to 2.25 meters. In this 7 link setup, the worst performance drop is in link 1 - this link is the 'bottleneck' - remember, the setup is event-based and only node 2 receives data packets at timed intervals. The maximum distance between 2 nodes was calculated to be 2.25 meters. Thus, node 2 suffers from node 1, node 3 and node 4. This is the same as in the previous experiment, but in this case, performance also suffers from the other links in the network, as can be seen in Figure 5.12, where node 8 suffers from node 6 and node 7. Therefore, performance is even worse than the previous experiment.

#### 5.3.3.4 Static routing backbone node configuration

This experiment's goal is to recreate the setup as the one that will eventually be used inside the train environment, with all nodes under full pressure. The performance figures

this setup produces will be used to compare with the required values, as were derived in Section 3.2.1.1 to see if the backbone configuration is actually suited for this type of application.
This section will continue with the experiment setup, results and conclusions.

1. The experiment is set up in the following way. Figure 5.14 shows the experiment setup, with a circle of radius $r = 2.25$ meter representing the transmission range of node 3.

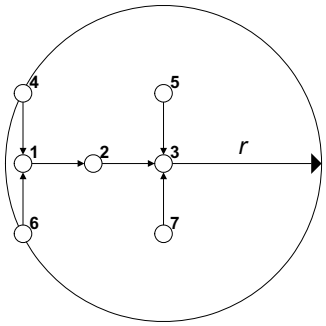
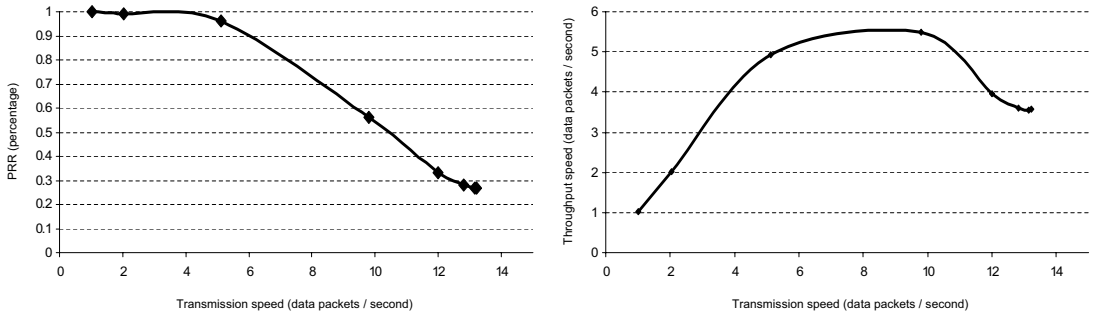



Figure 5.14: Backbone experiment setup

   (a) All nodes are Crossbow Mica2 nodes, equipped with a CC1000 transceiver.
   (b) A total of 7 nodes is used, with ID's ranging from 1 to 7.
   (c) The nodes with ID 1 up to 3 create the linear backbone array network.
   (d) The nodes with ID 4 up to 7 each represent an application, that sends out data to one node in the liner backbone array network.
   (e) The application nodes each transmit a data packet with a transmission speed of 1 packet / second - as was derived in Section 3.2.1.1.
   (f) The backbone nodes only forward data once they received data from an application node. They do *not* forward a data on regular basis, only when an application data packet has been received.
   (g) The environment used is the well-known office environment described in Section 5.1.
   (h) The static routing functionality, provided by TinyOS, is used.

2. The experiment results are the following. For a graph, see Figure 5.15.

   Figure 5.15(a) shows the transmission test results for the setup shown in Figure 5.14. The x-axis shows the actual transmission speed of node 1 and the y-axis the PRR of node 3.
   Figure 5.15(b) shows the throughput test results for the setup shown in Figure 5.14. The x-axis shows the actual transmission speed of node 1 and the y-axis the




(a) Transmission test results

(b) Throughput test results

Figure 5.15: Test results in backbone node setup

actual throughput speed, measured at node 38.
This figure closely resembles the results from Section 5.3.3.3 and yet again, performance has deteriorated. The maximum throughput rate of 5.4 data packets per second is, once again, not reached at the maximum transmission rate. Clearly, the MAC layer's performance suffers from the larger number whose data it has to accept - processing the data surely consumes time and performance. Fortunately, this configuration shows that it can reach a performance level of 4 data packets per second with a low loss rate.

3. This experiment shows that in a static environment the nodes are capable of being used for the backbone node configuration. Performance is again far worse than the previous experiment - as Figure 5.14 shows, node 3 suffers from all nodes involved in this experiment - node 4 and 6 are both $\sqrt{1^2 + 2^2} = \sqrt{5} = 2.24$ meter apart from node 3, which is less than the 2.25 meter calculated (which was still conservative). Summarizing, the CSMA/CA nodes can cope with data rates of up to 4 data packets per second coming from different regions, if programmed to the maximum throughput rate.

#### 5.3.3.5 Metric based backbone node configuration

This experiment's goal is exactly the same as the experiment conducted in Section 5.3.3.4 - only for this time, both geographic and metric based routing are used. The used MAC-layer is the standard TinyOS CSMA/CA protocol.

1. The experiment's setup is basically the same as the experiment from Section 5.3.3.4, only this time the metric based routing protocol from TinyOS is used, to be able to compare the geographic routing protocol with this metric based protocol. The MAC-layer involved is - in both cases - the standard TinyOS CSMA/CA-protocol.

2. The experiment results are the following. For a graph, see Figure 5.16 - this figure shows the routing method on the x-axis and the number of hops on the y-axis. Experiment results show that the throughput figure was roughly the same as the

figure in the experiment in Section 5.3.3.4. Besides this, the number of hops needed has dropped from 3 (static routing) to 1 for the metric based routing. Please note, that the transmission power was fixed to the maximum value - thus, the maximum distance data packets should be able to travel is far more than the 2.25 meters derived in Section 5.2.2.2. Data packets are transmitted to node immediately, instead of traveling through the nodes in between.

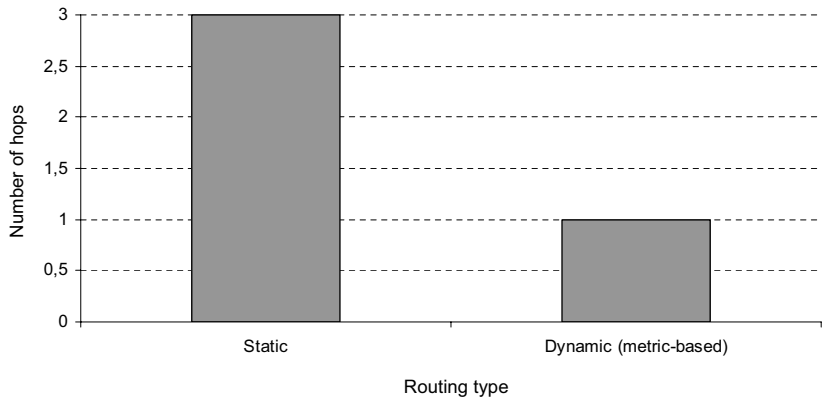



Figure 5.16: Comparison of static and dynamic test results in backbone node setup

3. The conclusions that can be drawn from this experiment are the following. The metric based routing protocol achieves the same performance level as the static routing protocol, while it uses a lower number of hops - clearly, this is the better choice.

## 5.4 Experimental evaluation conclusion

This chapter ends with a discussion on conclusions that can be drawn from this chapter. This chapter evaluated the selected reliability solutions and mechanisms from Chapter 4.
Section 5.1 discussed the experiment environment, which a static environment - an office room of the Logica building in Rotterdam, without any influences from other people, cellular phones etc. Initially, the intention was to perform the experiments in both a static and a dynamic environment. Unfortunately, there was no time to evaluate the dynamic environment.
Section 5.2 evaluated the link quality inside this static environment by analyzing the variation of both PRR and RSSI over distance. The path loss exponent $\eta$ for the static experiment environment is equal to 4,24, which is a good value for an office space - normally, the exponent lies between 4 and 6 in an office environment. The transitional region starts at a calculated distance of 2,25 meters. The formulas described in Section 2.2 turned out to be very useful and match the results very well.
Section 5.3 evaluated the performance level of the nodes inside this very same static

environment by looking at the data link layer's MAC mechanism and the network layer's routing mechanism, using various configurations.
The data link layer's MAC evaluation showed the less complex CSMA/CA MAC protocol is the better choice. The required performance rate is too high for the more complex Class 2 MAC protocols, even though reliability rates were decreasing dramatically over higher throughput rates in Class 1 protocols. Reliability rates however were worse than the Class 2 protocol. The maximum throughput rate of 5,4 data packets per second is not reached at the maximum transmission rate, but at a lower one. The network layer evaluation showed the metric-based protocol is the best routing protocol choice, since it uses the least amount of network hops. The metric-based protocol uses only 1 hop, compared to 3 hops for the static routing protocol.

# 6 Conclusions

This thesis project discussed the reliability and performance evaluation of a sensor network inside a train environment, that will be used as a data transportation backbone. This data transportation backbone will be used to attach several other train application sensor networks to. First, the background information chapter introduced relevant sensor network terminology, defined reliability and introduced relevant protocol layers protocols focused on and expanding reliability. Second, a requirement analysis was made, which showed which applications will use the data transportation backbone. The non-functional requirements showed the resource constraint and performance requirements, whereas the functional requirements showed the reliability and protocol layer requirements. Third, the solutions were selected that best matched the requirements. Finally, the selected solutions were evaluated.
This chapter will continue with a summary of the project conclusions in Section 6.1. Next, Section 6.2 presents the evaluation of this project. Finally, this chapter ends with a discussion of interesting subjects for future research in Section 6.3.

## 6.1 Summary

This section presents a summary of the background information, requirement analysis, solution selection and experimental evaluation chapter.

### 6.1.1 Background information summary

Chapter 2 presented the background information necessary to understand this project. A sensor network is a network of small sensor nodes with limited capacities, which can be of different capacity-types (therefor the name 'hybrid'). Because of these limited capacities, a node has a limited bandwidth, a limited amount of energy available and only small computational powers. The self-organizing functionality of the nodes offers the nodes ways to connect to each other according to a specific topology and thereby create a sensor network.
The environment the nodes are deployed in plays an enormous role. Reliability experiments using the most commonly used reliability metrics (being Packet Reception Rate and Received Signal Strength Indicator) showed undeterminable behavior - a phenomenon called 'radio-irregularity'. This means there are 3 regions in a link between 2 nodes: the connected region (PRR of 100 %), the transitional region (undeterminable PRR) and the non-connected region (PRR of 0 %). Fortunately, there are theoretical models available that model and explain the existence of these 3 regions using the PRR and RSSI metrics.
Reliability can be improved on both the data link and network layer by trying to solve

the hidden and exposed node problem. Therefor, a categorization is made in 3 MAC classes. On the network layer, the protocol solutions improve reliability by a specific routing method. Therefor, a categorization is made in several routing type classes, that all improve reliability in their own way.

### 6.1.2 Requirements analysis summary

Chapter 3 analyzed the applications that will use the backbone network for data transport and serve as a basis for the requirement analysis.
The train application analysis showed all nodes have fixed positions and there are 2 types of applications that will use the backbone nodes - condition monitoring and customer information applications. Each application has its own required minimum throughput rate and produces a continuous data stream. The application nodes will use the battery-powered low-capability nodes. The backbone nodes however will be wired high-capability nodes. Since the wired backbone nodes are the topic of this thesis, there is no need for energy savings.
All applications require a backbone node minimum throughput rate of 63 data packets per minute. To evaluate this required data rate, a series of setup configurations will be used to find out what the performance level of the backbone node configuration and the influence of the applications is.
Even though the final tests must be in a dynamic environment, the best way to start is by performing tests in a static environment to exclude external influences. To find out what reliability means, a profile must be made of the used environment that analyzes the transitional region and calculates the path loss exponent by using the PRR and RSSI metrics.

### 6.1.3 Solution selection summary

Chapter 4 discussed how to fulfill the requirements from Chapter 3, by matching the requirements from Chapter 3 with the available background solutions from Chapter 2.
The reliability calculation solutions focus on calculating and deriving the PRR and RSSI. All together, both of these items should not take much effort.
The network layer solutions focus on selecting the best suitable routing protocol type. Both the static routing and metric-based routing types were selected. The static routing protocol was selected for its simplicity and available (programming library) functionality and documentation. The metric-based routing was selected for its focus on reliability and also its available (programming library) functionality and documentation.
The data link layer solutions discussion in Section 4.3 selected both Class 1 and Class 2 protocols. The selected Class 1 protocol is the CSMA/CA protocol and the selected Class 2 protocol is the S-MAC protocol. The Class 3 protocol was found to be too complex and too badly-documented, even though it scored best on reliability. Therefor, no Class 3 protocol was selected.

### 6.1.4 Experiment evaluation summary

Chapter 5 evaluated the selected reliability solutions and mechanisms from Chapter 4. The experiment environment is a static environment - an office room of the Logica building in Rotterdam, without any influences from other people, cellular phones etc. Due to time limitations, there was no time to evaluate the dynamic environment.
The link quality evaluation inside the static environment analyzed the variation of both PRR and RSSI over distance. The path loss exponent $\eta$ for the static experiment environment is equal to 4,24, which is a good value - in general, the path loss exponent in an office space lies between 4 and 6. The transitional region starts at a calculated distance of 2,25 meters. Therefor, the maximum distance between nodes in with the used experiment setup must not be any higher than 2,25 meters.
The performance level evaluation inside this very same static environment looked at the data link layer's MAC mechanism and the network layer's routing mechanism, using various configurations.
The data link layer's MAC evaluation showed the less complex CSMA/CA MAC protocol is the better choice. The required performance rate is too high for the more complex Class 2 MAC protocols, even though reliability rates were decreasing dramatically over higher throughput rates in Class 1 protocols. The network layer evaluation showed the metric-based protocol is the best routing protocol choice, since it uses the least amount of network hops. The metric-based protocol uses only 1 hop, compared to 3 hops for the static routing protocol.

## 6.2 Project evaluation

This section evaluates this project by revising the original project description, presenting a renewed research question and goals and by evaluating the main contributions.

### 6.2.1 Revising project description

The original problem statement was the following.
Logica accepted an assignment by the Dutch Railways corporation - NS - to deploy a sensor network inside a train environment. However, during previous prototype experiments, Logica engineers experienced differing data delivery test results. This created a series of graduation projects, in which this thesis project is used to deploy a sensor network inside a train, as was described in Section 1.1.
This problem statement lead to the following central research question:

> *How can a reliable hybrid sensor network be established?*

The initial project goals, defined in Section 1.2, were the following.

1. Hybrid sensor support: the sensor network must support sensor nodes that have low, transmitting-only capabilities and high, both transmitting and receiving capabilities.

2. Improve robustness sensor network: the network must be made as immune as possible to possible disruptions.

3. Reliable data transport: mechanisms must be developed to transport data with the highest reliability.

4. Network diagnosis: the sensor network must be able to diagnose itself and report problems.

5. Data encryption: the sensor network must use a simple way of encryption to protect its data.

6. Evaluation: evaluation must take place in a dynamic train environment.

Due to the large amount of time it would take to achieve all these goals and answer the central research question, the project had to be scoped further more to make sure this project would have an ending in a considerate amount of time. Therefore, the problem statement was narrowed down to the following central research question:

> *How can a reliable data transport sensor network be established in a static environment?*

When comparing this research question with the original research question, goal 3 was fulfilled, but the focus on *hybrid* sensor networks (goal 1) and evaluating the network inside the dynamic environment (goal 6) were dropped, together with goals 2, 4, 5. This may look like dropping the most interesting subjects in this thesis project, but now, the emphasis came to lie on the basis of this project: reliable data transport and evaluating reliability in a sensor network application, which alone was enough to fill this masters thesis up to this point. This renewed research question led to a new series of goals, focused much more on data transport reliability in sensor networking.

1. To define reliable data transport in a sensor network.

2. To define a way of measuring data transport reliability.

3. To find out what the influences of used sensor nodes are on data transport reliability.

4. To find out what the influences of the implemented mechanisms are on data transport reliability.

5. To evaluate solutions and find out what their contributions are to data transport reliability.

6. To find out what the data transport reliability bottlenecks are in a deployed sensor network.

7. To find out what the influences of the train applications are on data transport reliability.

First, the renewed research question will be answered, followed by a discussion on the main contributions of this thesis.

### 6.2.2 Renewed research question

> *How can a reliable data transport sensor network be established in a static environment?*

A reliable data transport sensor network is established by knowing what data transport reliability means (goals 1 and 2), knowing the used sensor nodes (goals 3, 4, 5 and 6) and by knowing the sensor network application (goal 7).

1. Calculating data transport reliability itself is done with a simple metric - the PRR metric in Theorem 2.1. The difficult part lies in how to achieve the highest data transport reliability and what to expect of the results: when is a sensor network reliable? Therefor, the backgrounds of radio propagation is presented in Section 2.2, which showed there are 3 regions: the connected region, the transitional region and the non-connected region. Achieving the highest data transport reliability is done by performing experiments in the to-be-tested environments and evaluating both thePRR and RSSI values. Using the theoretical models on reliability from Section 2.2.5, this will show what the maximum distance is nodes can be apart with the highest PRR - without being in the transitional region.

2. The used sensor nodes play an enormous role in this project, which was shown in Chapter 5 - the evaluation chapter. Data transport reliability was evaluated by looking at the performance level of the nodes (Section 5.3) and by evaluating the radio model discussed in Section 5.2. The performance evaluation showed that even though simple protocols achieve less reliability, they are still the better choice due to the (too low) throughput rates of the more complex protocols. The radio model evaluation showed the enormous impact of the nodes' antenna and the way it propagates radio waves on reliability by finding out at which distance the transitional region starts.

3. Last but not least is the role of the train applications, which were discussed in Section 3.1. The backbone has to be able to transport all data these applications produce in time. A sensor network must be used for the correct applications - which means, low-bandwidth. The backbone configuration is on the edge of the nodes' maximum performance level. Wanting data faster simply will not work, especially not with the high demands on the protocols used - the higher the complexity, the lower the transport rate available. In high-bandwidth applications, transport reliability can be achieved easier by programming a node to transmit faster than by implementing a complex protocol.

With all these 3 items kept in mind, a reliable sensor network *can* be established in a static network with a predictable outcome - within a margin, of course.

### 6.2.3 Main contributions

This section presents the main contributions this thesis project has delivered.

1. First of all, this thesis report presented easy-to-use metrics to calculate the current data transport reliability of the equipment currently used by Logica, which are PRR and RSSI, together with solutions for the problems that occurred while trying to extract this data. These metrics deliver the possibility to quantify not only the quality of a single link but the quality of a specific environment as well.

2. Second of all, various radio wave propagation phenomena are discussed and explained, like for example the existence of the transitional region.

3. Third, the modeling and evaluation of radio wave propagation, which turn out to lie quite close to each other in a static environment. This method allows Logica engineers to calculate the maximum distance between nodes, without expecting any data transport problems.

4. Fourth, evaluating the performance level of the Logica sensor nodes - the nodes' capacities turn out to be easily reached in the currently used applications.

5. Last but not least, the evaluation of several reliability solutions: clearly, there is a difference between theory and practice; the performance evaluation showed there is no need for complex solutions, only for simple, high performance ones.

## 6.3 Future research

This section describes what the subjects of future research could be, when this project is continued by another person.
The original project goals were found to be far too large to be achieved by one single person in the time described in the project assignment. Therefor, there are still several interesting subjects left worth researching. These subjects are discussed in Section 6.3.1.
Second of all, a discussion on the train applications. In this thesis project, they were represented as one single node, but in real life they will be implemented by using a sensor network for a single application. These subjects are discussed in Section 6.3.2.
Third, possible subjects on the used hard- and software in Section 6.3.3. In this thesis project, the Crossbow Mica2 nodes were used together with TinyOS 1.x. However, technological inventions continue.
Fourth, the environment used to evaluate the backbone configuration in. This thesis used a static environment, but using a dynamic train environment poses many new subjects. These subjects are discussed in Section 6.3.4.
Finally, Section 6.3.4 discusses the monitoring of a complete train, which is a step further than just one single train compartment. This subject is discussed in Section 6.3.5.

### 6.3.1 Original project goals

Section 1.2 presented the original project goals. Of this list, goal 3 on reliable data transport was solved. This leaves the following goals.

1. Hybrid sensor support: the sensor network must support sensor nodes that have transmitting capacities only.

2. Improve robustness sensor network: the network must be made as immune as possible to possible disruptions.

3. Network diagnosis: the sensor network must be able to diagnose itself and report problems.

4. Data encryption: the sensor network must use a simple way of encryption to protect its data.

5. Evaluation: evaluation must take place in the dynamic train environment; see the discussion in Section 6.3.4.

All these items present interesting subjects, where each item is sufficient to fill one single thesis project - or even more.

### 6.3.2 Application nodes

As was stated before, the application nodes were represented by one single node in this thesis project. However, when the application nodes are deployed in real life, more than 1 single node will be used - possibly tens or hundreds of nodes. Since these nodes are the low-capability, battery-powered nodes, this poses new questions to reliability - all of a sudden, the problem of energy-wear out occurs! New reliability solutions will have to start focusing on energy savings as well to e.g. maximize the life time of these battery-powered sensor nodes.

### 6.3.3 Used hard- and software

The discussion on used hard- and software focuses on the usage of the Mica2 sensor nodes and the used software.

#### 6.3.3.1 Used sensor nodes

In general, the sensor nodes are too fragile to be programmed frequently. The 51-pin expansion connector, shown in Figure A.2(b), is used to connect the node to the programming board. Unfortunately, this done in such an instable way that a node will break down eventually. Besides this, the power connector is too fragile too and they are really hard to get hold of. Therefor, a possible next project would consist of the usage of other, more robust sensor nodes. Of course, this alone is not enough to start a new project, but this can be combined with the next remarks on TinyOS.

#### 6.3.3.2 Used software

During this project, the used version of TinyOS was 1.1.x, since all previous students had used this version. However, currently, TinyOS version 2.x is available, which poses some interesting options, such as over-the-air programming, which saves a lot of time - certainly in applications with large node numbers. However, version 1.1.x and 2.x code is not exchangeable. An interesting project subject would be to use the newer version, in combination with other, more robust sensor nodes.

Besides this, there is also the layered mechanisms that can be further improved - for example, especially in the backbone nodes, data aggregation can be useful - many data has the same origin, so why not combine this to reduce the overall transport data?

### 6.3.4 Dynamic train environment

All evaluation in this thesis project took place in a static environment, to find out what the influences and test results of reliability and several mechanisms were. If these tests were done in a dynamic environment, it would not have been clear if the results were due to the environment or due to the implemented mechanisms. Next step is to perform all tests in a dynamic environment, with moving objects and obstacles in the way.

### 6.3.5 Complete train

In this thesis project a single train compartment was monitored. An interesting next subject would be to monitor a complete train - this also raises the problem of how to connect the train compartments together to one single sensor network.
A next step, coming after the monitoring of a complete train, is that of the influence of other trains - if trains pass each other, what will happen? Or what if the train is in the station? These are all interesting subjects worth investigating.

# Appendices

# A

This chapter discusses several information items that are relevant to this project, but are too long to be included in the main chapters. The first item presents background information on Logica in Section A.1. Second, Section A.2 discusses the Crossbow Mica2 sensor nodes used. Third, the programs and files used are presented in Section A.3. Finally, Section A.4 presents background information on the Open Systems Interconnection (OSI) model.

## A.1 Logica

This graduation project is done at Logica - formerly known as LogicaCMG - in Rotterdam. LogicaCMG plc. is the result of the merger of Logica plc. (60 %) with CMG plc. (40 %) on December 30, 2002. Both service providers originate from the UK, but CMG Netherlands was substantially larger than CMG UK. In February 2008, LogicaCMG changed its name to Logica and Logica Rotterdam moved from the Kralingseweg to the George Hintzenweg.
Logica is a major international force in IT services and employs 40000 people across 41 countries. Logica's focus is on enabling its customers to build and maintain leadership using Logica's knowledge. Figure A.1 shows all of Logica's divisions, focusing on different markets. This project is placed in the IDT division. Each division is divided in several competences. This project is placed in the Technical Software Engineering 1 (TSE1) competence.

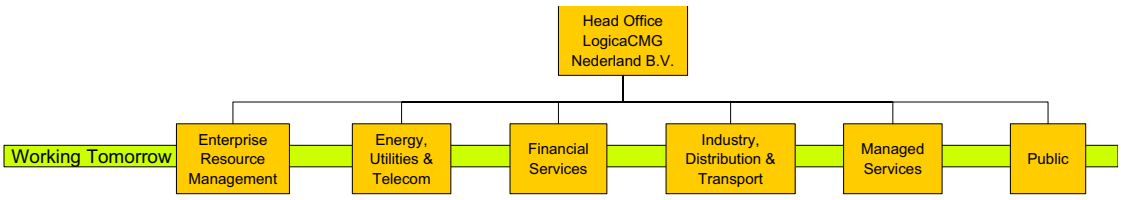
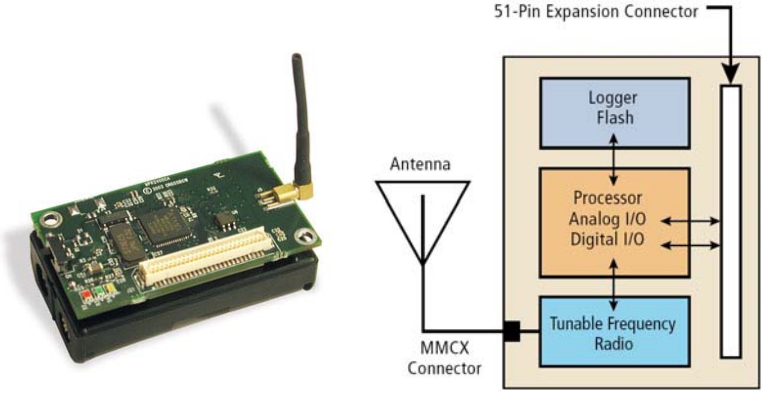



Figure A.1: Logica organization structure with Working Tomorrow

Graduation projects are placed in the Working Tomorrow (WT) program. This program is a coöperation between all the divisions shown in Figure A.1. Within the WT program, students graduate on reviewing technological developments on feasibility and possibilities. They develop prototypes to present new insights into new technology. Besides this, student have the possibility to do a project in a commercial surrounding. Last but not least, with the WT program, Logica emphasizes its innovative character and gets the chance to find future employees.

## A.2 Mica2 sensor nodes

This section discusses the backgrounds on the sensor nodes used, which are the Mica2 nodes by Crossbow [6]. Figure A.2(a) [6] from Section 2.1.1 shows a figure of a Mica2 sensor node, which is repeated here for convenience. Figure A.2(b) shows the internal structure of the Mica2 sensor node, which unfortunately does not exactly match the node shown in Figure A.2(a). The internal structure consists of the 51-pin expansion connec-




(a) Outside image [6] (b) Internal structure [6]

Figure A.2: Overview Mica2 sensor node

tor, flash memory, the processor, the tunable frequency radio, the MMCX connector and the antenna. All data originates from the Mica2 data sheet [6].

1. The 51-pin expansion connector is used for external inputs and supports analog inputs, digital I/O, I2C, SPI and UART interfaces. The UART interface is used to program the node with a programming board.

2. The program flash memory is used for storing data and has a size of 128 kBytes. Besides this, it has 512 kBytes of (serial) flash memory for measurements and a 10 bit Analog to Digital Converter (ADC) converter. The 10 bit ADC converter is used for calculating the RSSI value as discussed in Section 2.2.1.2.

3. The processor integrated on the Mica2 node is based on the Atmel ATmega128L.

4. The tunable frequency radio - the transceiver - is a ChipCon (CC)1000, which is programmed to function on 868 and 916 MHz.

5. The MMCX connector connects the antenna with the sensor node PCB. MMCX is a series of industrial connectors.

6. The antenna speaks for itself.

The nodes run their own OS, TinyOS. For more information on TinyOS, see Section A.3.1.

## A.3 Programs and files used

This section discusses both the programs and files used throughout this project. Programs used are discussed in Section A.3.1, files used are discussed in Section A.3.2.

### A.3.1 Programs used

This section discusses the computer programs used throughout this project. This list does not include commonly-used software such as Microsoft Windows, Office et cetera, but it does include Cygwin and TinyOS.

1. Tiny Operating System (TinyOS) is a free and open source component-based operating system, which targets at WSNs. TinyOS applications are written in nesC, which is a dialect of the C programming language optimized sensor networks and its limitations. Libraries and tools - such as the nesC compiler - are mostly written in C. The TinyOS programs are built from software components - some of them represent hardware abstractions. Software components are connected together using interfaces. Common abstractions for e.g. packet communication are provided by TinyOS. The current version - since July 2007 - is 2.0.2, but in this project the also still commonly-used version 1.15 is used, since programs made by previous Logica graduation students are written in version 1.15. Version 2.x-code is not interchangeable with version 1.x-code, due to its different structure.

2. Cygwin is a collection of tools that run on Microsoft Windows, that behave in ways familiar to Unix users. Example programs are GNU Compiler Collection (GCC), a collection of compilers and VI, a screen-oriented text editor. Cygwin is also included in the TinyOS distribution.

### A.3.2 Files used

The files described in this thesis report that need explanation are AM.h and SMACTest.

1. The file 'AM.h' is a header file, showing the structure of a data packet transmitted. After installing the TinyOS Windows executable file, the file 'AM.h' can be found in the directory ROOT/tinyos-1.x/tos/types, where 'ROOT' is the Cygwin installation directory.

2. The 'SMACTest' program is part of the S-MAC test suite, which can be downloaded for free from [26] and includes test programs using the S-MAC protocol.

## A.4 OSI model

This section discusses the Open Systems Interconnection (OSI) model [2, 12]. The OSI model is a layered, abstract description for communications and computer network protocol design, consisting of seven layers. Table A.1 shows the seven layers of the OSI model, together with their respective data units. First, Section A.4.1 discusses the function of each layer. Following this, Section A.4.2 discusses the layers applicable to this project.

| Layer type | Data unit | Layer description |
|---|---|---|
| Host layers | Data | 7. Application |
| | | 6. Presentation |
| | | 5. Session |
| | Segments | 4. Transport |
| Media layers | Packets | 3. Network |
| | Frames | 2. Data link |
| | Bits | 1. Physical |

Table A.1: Layers in the OSI model [2, 12]

### A.4.1 Layers in OSI model

This section serves as a short introduction into the seven OSI-model layers. These layers are on display in Table A.1, together with the data units used in that specific layer.The seven layers are divided in host layers and media layers.

#### A.4.1.1 Host layers

Host layers are specific for one device on the network. The host layers are the application, presentation, session and transport layer. Layers 7 to 5 are dealt with by the Operating System (OS), whereas Layer 4 is the interface between the device and the network.

7. The application layer interfaces applications and provides access to lower layer functions and services.

6. The presentation layer negotiates syntactic representation and performs data transformations, e.g. compression and code conversion.

5. The session layer coordinates connection and interaction between applications and it establishes dialog, manages and synchronizes data flow direction.

4. The transport layer provides mechanisms for establishing a connection between users, while supporting Quality-of-Service (QoS) mechanisms. Quality-of-Service is used to express the quality of of the wireless sensor network, which in this project is link quality.
   The transport layer interfaces the top-three layers from the bottom-three layers. An example transport layer protocol is FTP.

#### A.4.1.2 Media layers

Media layers define behavior involving all devices on the network. These layers are the network, data link and physical layer.

3. The network layer provides mechanisms for routing the data units across the network. An example network layer protocol is TCP.

2. The data link layer provides mechanisms for accessing the shared physical medium by the device. An example data link layer protocol is IP.

1. The physical layer delimits and encodes the bits onto the physical medium. It defines the electrical, mechanical end procedural formats. An example protocol is RS232.

### A.4.2 Reliability in layers

This section discusses which OSI layers are applicable to this project. First, Layer 1, which is the interface to the shared medium. This interface is defined by the transceiver and its (logical) implementation into the sensor node. The transceiver design and implementation plays a large role in reliability - for example, is the transceiver transmitting with the same amount of power in each direction? - but this is not something that can be changed, since the antenna is already implemented in the design of the sensor nodes. The data link layer is the first layer that actually deals with the problem of accessing the shared medium. Multiple nodes transmitting at the same moment leads to collisions. This results in packet loss and a decrease of reliability. The data link layer mechanisms can either prevent these problems or let them happen - therefor, they play a large role in reliability.
The network layer makes sure data packets actually arrive at the intended destination and defines which route must be used. This is perhaps the most obvious layer that helps in achieving reliability.
All other (upper) layers focus on service-oriented functionality and presentation. Of course, service-oriented functionality can be used to improve reliability, but this will still be based on Layer 2 and Layer 3 functionality - the layers that are responsible for and influence the current reliability rate. Therefor, the layers applicable to this project are Layer 2 and 3.

# Bibliography


[1] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In Proceedings of the first international conference on Embedded networked sensor systems, pages 1–13, 2003.

[2] Zimmermann, H. OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection. IEEE Transactions on Communications. 1980.

[3] G. Zhou, T. He, S. Krishnamurthy and John A. Stankovic. Impact of radio irregularity on wireless sensor networks. Proceedings of the 2nd international conference on Mobile systems, applications, and services. 2004.

[4] M. Zuniga and B. Krishnamachari. Analyzing the transitional region in low power wireless links. 2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.

[5] H. Nikookar amd H. Hashemi. Statistical modeling of signal amplitude fading of indoor radiopropagation channels. Universal Personal Communications, 1993.

[6] Crossbow MICA2 product page. `http://www.xbow.com`.

[7] B. Krishnamachari. Networking Wireless Sensors. Section 6.2.2. Cambridge University Press. 2005.

[8] Copyright owned by Logica Rotterdam.

[9] R. Hekmat. Ad-hoc Networks: Fundamental Properties and Network Topologies. Springer Books. 2006.

[10] A. Grama et. al. Introduction to parallel computing - Second edition. Addison Wesley. 2003.

[11] G. Finn. Routing and Addressing Problems in Large Metropolitan-Scale Internetworks. ISI Research Report ISU/RR-87180, March 1987.

[12] Wikipedia - OSI Model. `http://en.wikipedia.org/wiki/OSI_model`.

[13] D. B. Johnson et. al. DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks. Carnegie Mellon University. `http://www.monarch.cs.cmu.edu`.

[14] Wikipedia - Omnidirectional antenna. `http://en.wikipedia.org/wiki/Omnidirectional_antenna`.

[15] Deepak Ganesan, Ramesh Govindan, Scott Shenker, and Deborah Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. Mobile Computing and Communications Review, vol. 4, no. 5, October 2001. `http://citeseer.ist.psu.edu/article/ganesan01highlyresilient.html`.

[16] J. H. Chang et. al. Energy Conserving Routing in Wireless Ad-hoc Networks, Proceedings of IEEE INFOCOM, March 2000.

[17] Wikipedia - Energy Harvesting. `http://en.wikipedia.org/wiki/Energy_harvesting`.

[18] Wikipedia - Aloha. `http://en.wikipedia.org/wiki/ALOHAnet`.

[19] Computer Networks. A. Tanenbaum. Prentice Hall. 1981.

[20] L. Kleinrock and F. Tobagi. Packet switching in radio channels, part 1: Carrier sense multiple-access models and their throughput-delay characteristics. IEEE Transactions on Communications. 1975.

[21] Rahman, A and Gburzynski, P. Hidden Problems with the Hidden Node Problem. 23rd Biennial Symposium Communications, 2006. Page(s) 270 - 273. 2006.

[22] Yihong Zhou and S. M. Nettles. Balancing the hidden and exposed node problems with power control in CSMA/CA-based wireless networks. IEEE Wireless Communications and Networking Conference. 2005.

[23] Simon S. Lam. A Carrier Sense Multiple Access Protocol for Local Networks. University of Texas at Austin. 1979.

[24] Alec Woo and David E. Culler. A transmission control scheme for media access in sensor networks. Proceedings of the 7th annual international conference on Mobile computing and networking. 2001.

[25] TinyOS: Open Source Operating System for sensor networks. `http://www.tinyos.net`.

[26] S-MAC download library: `http://www.isi.edu/ilense/software/smac/`.

[27] P. M. Shankar. Introduction to Wireless Systems. Wiley, 2001.

[28] Z. Haas and J. Deng. Dual busy tone multiple access (DBTMA) - a multiple access control scheme for ad hoc networks. IEEE Transactions on Communications, volume 50, pp. 975985. June 2002.

[29] Wei Ye, John Heidemann, and Deborah Estrin. Medium access control with coordinated, adaptive sleeping for wireless sensor networks. Tech. Rep. ISI-TR-567, USC Information Sciences Institute, Jan. 2003.

[30] IEEE 802.11 WIRELESS LOCAL AREA NETWORKS - The Working Group for WLAN Standards. ANSI/IEEE Std 802.11, part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. IEEE 802 LAN/MAN Standards Committee, 1999.

# List of Acronyms

**4B6B** 4-bit / 6-bit

**ADC** Analog to Digital Converter

**dB** decibel

**CC** ChipCon

**CODA** COngestion Detection and Avoidance

**CPU** Central Processing Unit

**CRC** Cyclic Redundancy Check

**CSMA** Carrier Sense Medium Access

**CSMA/CA** Carrier Sense Medium Access with Collision Avoidance

**CSMA/CD** Carrier Sense Medium Access with Collision Detection

**CTS** Clear To Send

**DBTMA** Dual Busy Tone Multiple Access

**DSR** Dynamic Source Routing

**GCC** GNU Compiler Collection

**GRaD** Gradient cost routing

**IDT** Industry, Distribution & Transport

**LQE** Link Quality Estimator

**MAC** Medium Access Control

**MACA** Medium Access Collision Avoidance

**MSR** Multipath Soure Routing

**NS** Nederlandse Spoorwegen

**OS** Operating System

**OSI** Open Systems Interconnection

**PCB** Printed Circuit Board

**PRR** Packet Reception Rate

**PSFQ** Pump Slowly Fetch Quickly

**QoS** Quality-of-Service

**RF** Radio Frequency

**RAM** Random Access Memory

**ROM** Read-Only-Memory

**RSSI** Received Signal Strength Indicator

**RTS** Ready To Send

**S-MAC** Sensor-MAC

**SECDED** Single Error Correction, Double Error Detection

**SNR** Signal-to-Noise Rate

**T-MAC** Timeout-MAC

**TDMA** Time Division Multiple Access

**TinyOS** Tiny Operating System

**TSE1** Technical Software Engineering 1

**WSN** wireless sensor network

**WT** Working Tomorrow

# Curriculum Vitae

**Sander Bastiaan Kootkar** was born in Breda, the Netherlands, on March 20, 1983. He obtained his VWO diploma in 2001 at Markland College in Oudenbosch. In June 2004 he received his Bachelor of Engineering from Avans Hogeschool in Breda by completing the Electrical Engineering program. Sander joined the Computer Engineering department of Delft University of Technology in September 2004. He will graduate in July 2008 by completing his MSc. thesis *Reliable sensor networks: a case study in commuter trains*.