Lassi-Pekka Hirvonen

LPWA evaluation platform

Helsinki Metropolia University of Applied Sciences

Master of Engineering

Information Technology

Master's Thesis

10 May 2018

Helsinki Metropolia University of Applied Sciences

Author(s) Title Number of Pages Date	Lassi-Pekka Hirvonen LPWA evaluation platform 58 pages + 5 appendices 10 May 2018
Degree	Master of Engineering
Degree Programme	Information Technology
Specialisation option	Networking and Services
Instructor(s)	Risto Pajula, Embedded Systems Segment Manager Ville Jääskeläinen, Principal Lecturer

Today new era of connectivity is shaping the industry. Things are being connected to Internet which means billions of connections in worldwide perspective. These connections are utilizing communication technologies with high complexity. Networks consist of actuators which are operating usually in radio and TCP/IP environment. Reliability is the most important factor in industrial communication. Network sensors should be available with sufficient transmission delay. How to ensure that the communication system fulfils the applications requirements and high availability?

This thesis provides a practical approach for evaluating complex communication systems. An evaluation platform was designed and implemented in order to measure modern communication system key parameters and availability. Key parameters included extended quality of service measurement such as packet error ratio, signal to noise ratio, received signal strength, and delay measurements. The availability was analyzed with statistical analysis of transmission delays. Finally, measured delays were compared to calculated theoretical values. With extended quality of service, the flaws and weaknesses of the communication system can be analyzed. It is also possible to detect poor behaving network components or even verify the radio network design with radio channel measurements.

The focus of the thesis is on the transmission quality. The network access and security issues were left out from the study. Furthermore, the Long Range Wide Area Network (LoRaWAN) communication technology was chosen for the evaluated technology. The Evaluation platform architecture was designed so that other communication technologies can be easily added later on to the platform, otherwise rest of the Low Power Wide Area (LPWA technologies) are not included in this thesis.

Evaluation platform is utilizing Python math, graphic libraries and Raspberry pi mini computers. Evaluation output is presented by the Wapice Internet Of Things (IoT) solution. So the evaluation platform itself is an IoT solution which is analysing the performance data of its own network components.

Keywords	LPWA, LoRa, LoRaWAN, IoT, QoS, transmission delay

Table of Contents

List of Abbreviations

API	Application Program Interfaces
CDF	Cumulative Distribution Function
CRC	Cyclic Redundancy Check
DSNA	Diagnostic Server/Network Adapter
ECDF	Empirical Cumulative Density Function
ISM	Industrial, Scientific and Medical radio band
LoRaWAN	Long Range Wide Area Network
LPWA	Low Power Wide Area
LTE	Long Term Evolution
LTE Cat-M	Long Term Evolution Category M
LTE Cat-1	Long Term Evolution Category 1
MAC	Media Access Control
NB-IoT	Narrow Band Internet of Things
NB-LTE	Narrow Band Long Term Evolution
NTP	Network Time Servers
OTAA	Over the Air Activation
PDF	Probability Density function
PER	Packet Error Ratio
QoS	Quality of Service
RDU	Raspberry pi Diagnostic Unit
REST	REpresentational State Transfer
RPi	Raspberry Pi
RSSI	Received Signal Strength Indicator
SNR	Signal to Noise Ratio

1 Introduction

Internet of things is emerging with set of different communication technologies, which will shape the world in many ways. One of the newest long range technology in the market is the Long Range Wide Area Network (LoRaWAN) technology. LoRaWAN is actually a long range protocol which has been designed by LoRa alliance. This new long range protocol is used mainly in applications, which are producing a great amount of data from thousands of network end devices. According to the LoRaWAN alliance, the key application areas are smart cities, supply chain applications, intelligent building applications, connecting consumer devices and control of agriculture applications [1 pp.4].

Wapice is a technology partner, which is providing software expertise to industrial companies. As a leading technology service provider, Wapice is constantly looking for innovative solutions to fulfil the customer needs. In near future things will be connected to the Internet. The most common way to connect devices is Bluetooth based communication systems. On the other hand in terms of link distance, Bluetooth based solution performance does not typically fulfil application requirements. LoRaWAN based technology could be an answer to different long distance connectivity needs with low energy consumption.

As Figure 1 illustrates, LoRaWAN is one of the Low Power Wide Area (LPWA) technologies, which is competing for customers who are seeking a long distance communication link with a low data throughput [2 pp.2].

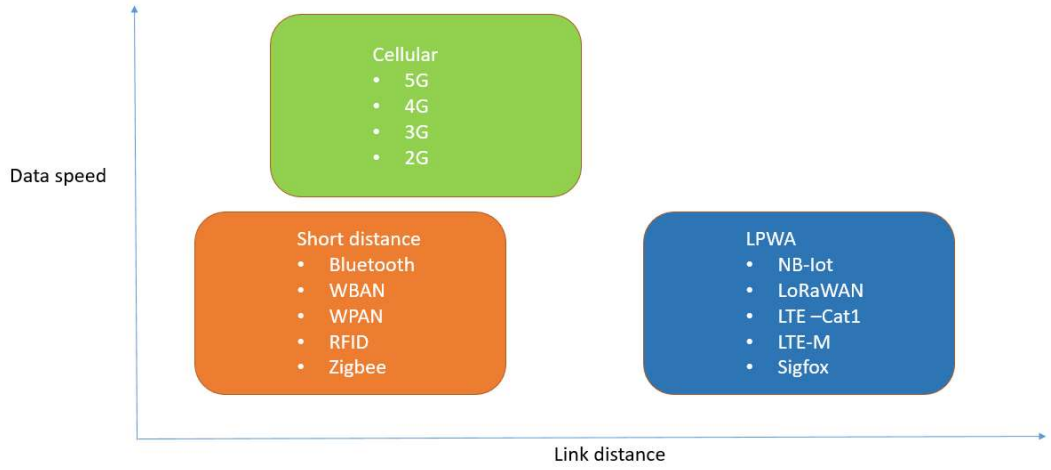

Figure 1. LPWA technologies.

According to the LoRa Alliance, the technology seems to fit a wide set of applications but is it truly so? It is not so easy to evaluate the technology in terms of a professional use. Network simulators can provide an estimation of latency and throughput but the availability time require tests with real LoRaWAN devices. In practice the network weakest link might degrade the system overall performance in many ways.

There seems to be many open source LoRaWAN solutions available in the market. How to ensure that the network components are suitable for a professional use. Does the LoRaWAN system constrains prevent the use of professional, mission critical applications? Because the communication system is as effective as its weakest link, how to evaluate the communication system reliability and the true field performance?

This thesis work is about solving this evaluation problem by providing a tool (a platform) for an overall system analysis. With this tool, the future LPWA studies can evaluate single network components or a network overall performance. The tool should be able to measure the most important communication system parameters. It should provide an availability, a latency, a packet error rate and network throughput measurements for a future LPWA network and its component studies.

The focus of this thesis is on measuring LPWA data transmission quality. The network access and security issues were left out from the study. Furthermore, the LoRaWAN communication technology was chosen for the evaluated technology, other LPWA technologies are out of the thesis scope. The focus of the evaluation platform is to generate and analyse measurement results with sufficient accuracy for most of the expected applications.

Thesis is divided into six chapters. The chapters 2 and 3 introduces the problem background theory and thesis metrics. The fourth chapter is about the solution model, in other words how the evaluation platform was conducted. The fifth chapter presents the measurement results and the final chapter consist of conclusion of the thesis work and evaluation of the platform performance.

2 Quality of Service in LPWA Technologies

According to LoRaWAN alliance, there are five different LPWA technologies today, which characteristics seems to differ greatly. As a conclusion from Table 1, the data rate improvement seems to require more power and vice versa. Naturally, battery lifetime is following the power efficiency performance. Narrow Band Long Term Evolution (NB-LTE) and LoRaWAN possess a similar performance. Other Long term Evolution (LTE) based technologies such as Long term evolution Category 1 (LTE Cat-1) and Long Term Evolution Category M (LTE Cat-M) has a significantly better data rate. [3 pp.4].

LTE Cat-1, LTE Cat-M, NB-Iot and Sigfox technologies are subscription based cellular systems, so the network infrastructure is owned by a network operator. Instead, LoRaWAN infrastructure can be also owned by the application provider.

The lowest data rate is achieved by Narrow-Band technology due to narrow bandwidth. Narrow radio channel means a low noise contribution and the receiver is able to detect transmissions with extremely low power levels. This proprietary technique is provided by Sigfox company and detailed specification are not available in public [4 pp.4].

LTE based LPWA technologies are divided into different categories. The initial version of the machine type of communication was based on LTE Cat-1. However this solution did not support low battery consumption, range and cost requirements. So LTE Cat-M was released in order to increase the range and reduce the power consumption with power reduction techniques. Following technology leap, the NB-LTE provided even better performance. NB-LTE is already replaced by Narrow Band Internet of things (NB-IoT) with new radio access. This enables improved coverage, support for massive number of low throughput devices, low delay, low device cost, low device power consumption and optimized network architecture [5 pp. 1-3].

Table 1. LPWA technologies comparison [3 pp.16].

Feature	LoraWAN	Sigfox	NB-IoT	LTE-cat1	LTE cat-M
Modulation	SS Chirp	UNB	QPSK	OFDMA	OFDMA
Link Budjet	154 Db	151 Db	150 dB	130 dB	146 dB
Battery life-time (2000 mAh)	105	90			18
Data rate	290 bps – 50 kbps	100 bps	DL:234.7 kbps; UL:204.8 kbps	10 Mbps	200 kbps – 1Mbps
Power effi-ciency	Very high	Very high	Medium high	Low	Medium
Interfer-ence im-munity	Very high	Low	Low	Medium	Medium

NB-IoT performance is similar to LoRaWAN in Figure 2 [2]. The most significant differences are presented in Figure 2. The latency performance is better in NB-IoT because there are no such duty-cycle limitations as in LoRaWAN. In addition, the NB-IoT physical layer is more efficient in terms of synchronization and latency.

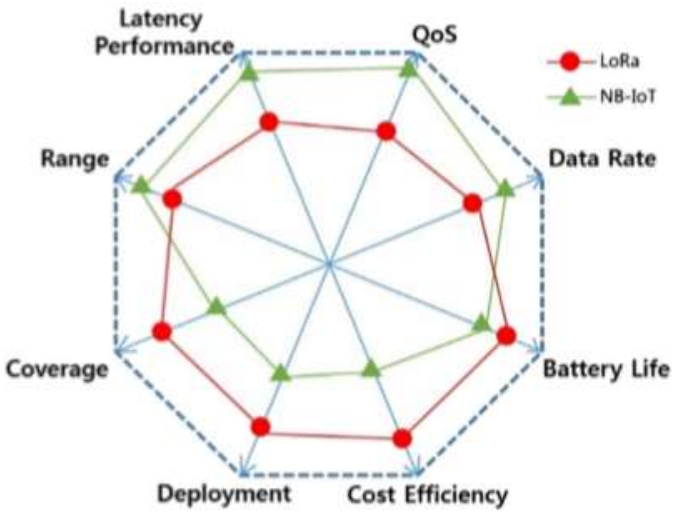

Figure 2. Differences between the LoRaWAN and NB-Iot [2 pp.20].

Table 1 is presenting a narrow perspective when different LPWA technologies are compared. For example the latency is neglected which is a crucial feature in many applications. Sorensen et al defined that the LoRaWAN is utilizing the unlicensed radio spectrum, which requires duty cycling [6 pp. 1]. The duty cycling sets limitations to data rate and latency. So it is not so easy to evaluate LPWA technologies due to complex nature of the systems, radio regulations and environment behavior.

This thesis work is concentrating on how to evaluate LPWA technologies, especially the LoRaWAN solution. This evaluation is based on LoRaWAN quality of service measurements which are presented in next chapter.

2.1 Quality of Service Requirements

Any telecommunication network is a complex entity, and its overall performance is depended on many different network parameters. The LPWA technologies consist of radio networks, which are generating randomness to data transmission. So how to measure network performance and what are the most critical parameters?

The Quality of Service (QoS) is well established practice for measuring transmission quality and service availability of networks. In order to implement an efficient quality of service, high availability is an essential requirement for a network. In addition, transmission quality is measured with packet loss, delay and delay jitter parameters [7].

2.1.1 High Availability

A high availability network should reach a 99.999 % uptime. This means 5 minutes downtime within a one year period. High availability is formulated from equation 1 [8].

$$Availability = \frac{MTTF}{MTBF} = \frac{MTTF}{MTTF+MTTR} \quad (1)$$

MTTF is a mean time to failure
MTBF is a mean time between failures
MTTR is a mean time to repair

System availability can also be defined as an availability of services (equation 2).

$$Service\ availability = \frac{Service\ uptime}{Service\ uptime + Service\ outage} \quad (2)$$

The availability definition seems to vary depending on the application. Some research project are measuring response time performance and some others uptime of the system [8 pp.56]. In this thesis, the availability is defined as a continuous measurement of uptime over the overall measurement time.

2.1.2 Transmission Quality

In the Figure 3 there is a presentation of a LoRaWAN QoS measurement which consist of a delay, loss and delay jitter measurements.

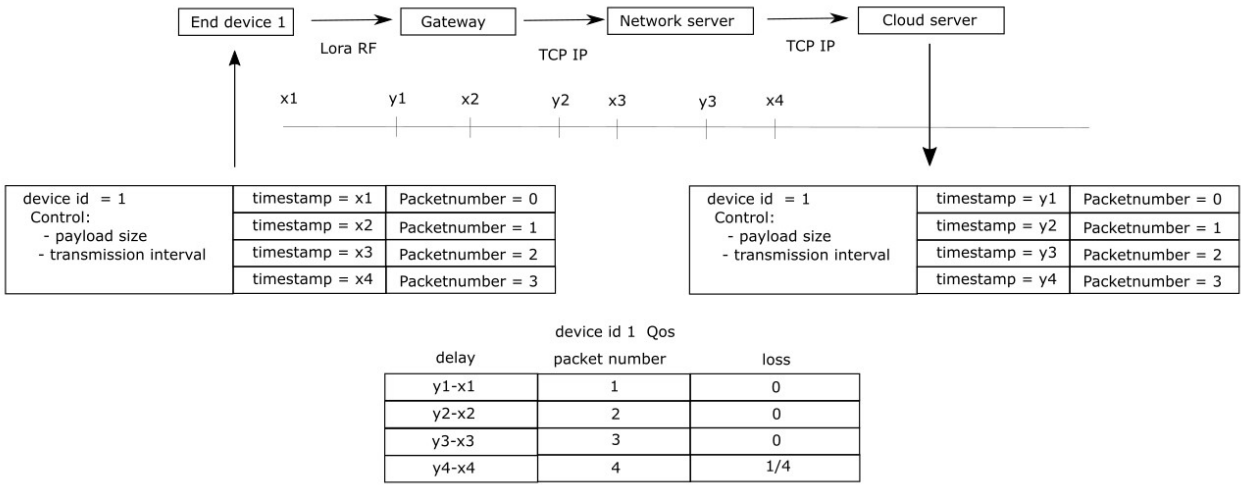

Figure 3. QoS in LoRaWAN.

The loss defines the lost data packets in transmission. In other words the amount of packets which were not received compared to total number of transmitted packets. In high availability networks the loss is almost zero. A delay measures the transmission time, the amount of time what takes a packet to travel from a transmitting endpoint to receiving endpoint. Jitter is the variation of the end to end delay [7]. In other word's jitter is a discrete distribution of transmission delay, so it is intuitive to analyze jitter with Cumulative Distribution Function (CDF) [9].

However, this measurement setup does not provide any information where the problem is when packets are missing or transmission delay is out of desired values. Therefore, QoS is a user perspective to quality, not a tool for solving communication issues.

2.1.3 Latencies in Communications Systems

Latency is defined different ways in telecommunication system. In network measurement, latency is a delay between two network nodes [10]. On the other hand, on other sources it is described as a delay between request and a response message [11].

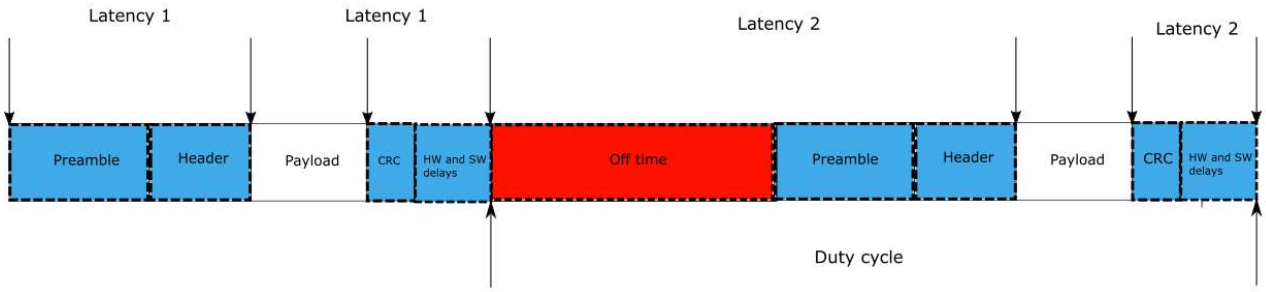

Figure 4. Latencies and duty cycle.

As Figure 4 illustrates, the latency case number 1 consists of preample, header, and CRC section. Naturally, some additional delay is caused by hardware and software implementation due to data has been processed and buffered.

In order to obey the duty cycle restriction the required off time has to be added to latency calculation as seen in Figure 4. Obviously, the cases in field measurement are not so simple because transmission is able to start in the middle of the off time. Or there can be other transmission, which might cause collisions and data retries which are seen as an additional latency in a measurement.

2.1.4 Synchronization in Timing Measurements

In order to achieve sufficient accuracy in QoS measurements, the measurement network should be synchronized to a common clock source. Servers that provide timing services are called Network Time Servers (NTP). A NTP network consist of stratum layers. Every layer synchronizes themselves on upstream of time servers [12]. In Figure 5, the layer 1 is synchronized directly to accurate GPS clock source so accuracy is high on the top.

Since the lower layers are synchronized to upstream via network connection, accuracy decreases on the way to the bottom.

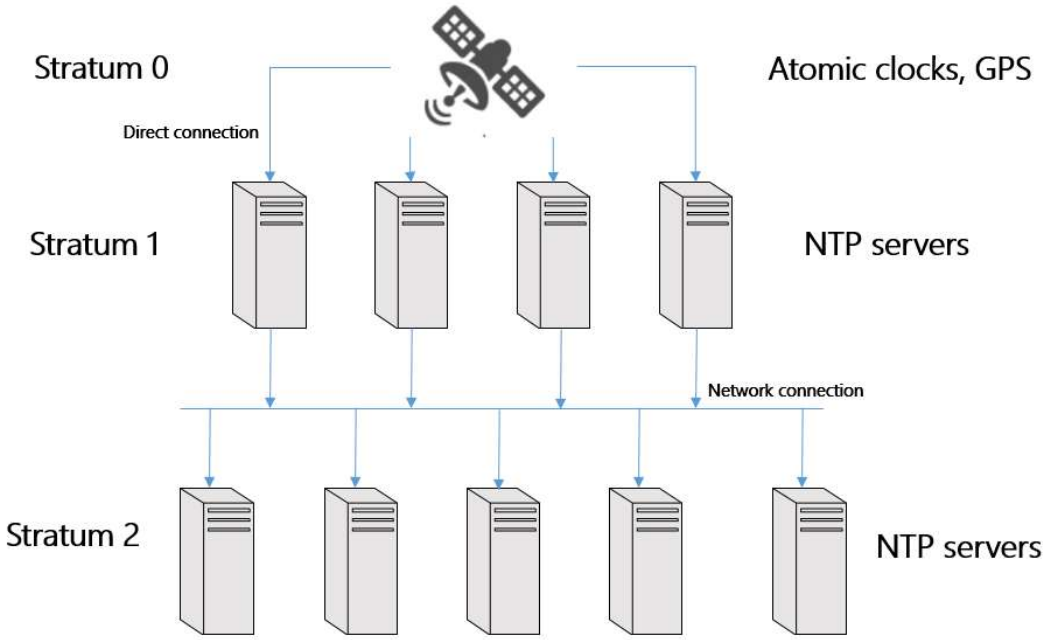

Figure 5. NTP server hierarchy.

Different applications and their latency requirements are listed in Figure 6 [13].

Application	Update time (s)	Latency (s)	Sync err. (s)	Data Size (B)	# Nodes
Industrial Automation (Factory)	<0.1	<0.01	$<10^{-4}$	<100	10^2
Industrial Automation (Process)	<60s	<1	<0.01	<100	10^3
Smart Building (e.g. HVAC)	<600	<60	<1	<1k	10^2
Home Automation (e.g. lighting)	Event based	<60	$<10^{-4}$	<100	10^2
Smart metering (Electricity)	<600	<10	<1	100	10^6
Smart metering (Gas)	4/day	<3600	<60	100	10^6
Smart Grid (PMU)	<0.01	<0.01	$<10^{-6}$	<200	10^2
Smart Grid (EV fleet)	4/day	<15	<60	<300	10^2

Figure 6. Latencies and data size in different applications [13].

On the latency point of view, the strictest requirement comes from Industrial automation applications. A measurement system should provide 1 ms accurary in latency measurement in order to measure timings, which are presented in Figure 6.

2.1.5 Using ECDF Function in Transmission Jitter Analysis

Analysing jitter is an important application of probability functions [9]. Jitter consists of discrete samples of the transmission delay. Figure 7 shows that due to this discrete nature, the probability density function is a histogram of delay values. The samples are scaled so that the sum of the samples yields to 1. Therefore, one can assume that the probability of having any result is 100% [9].

$$PDF(x_i) = \frac{1}{N_{tot}} \times N(x_i) \quad (3)$$

PDF is a relative probability of value X_i
X_i is a result
$N(x_i)$ is the frequency of value X_i
N_{tot} is a total number of samples

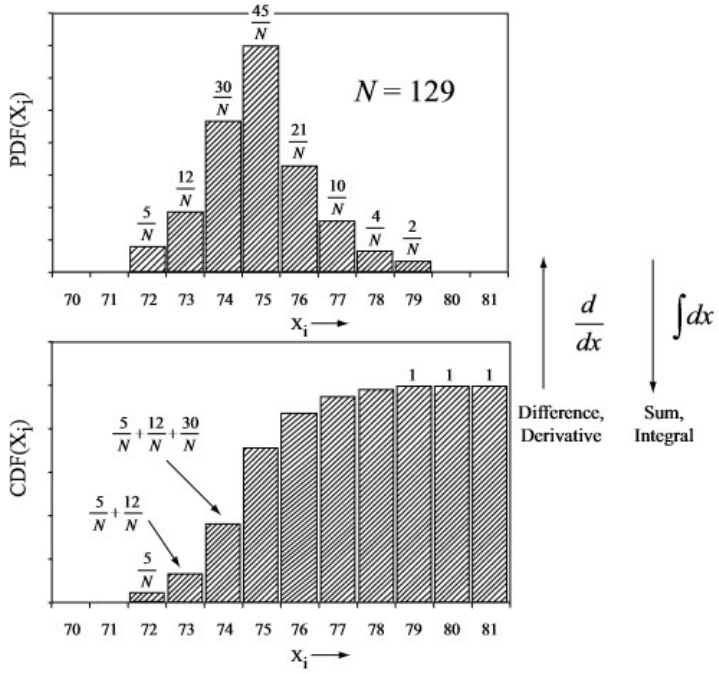

Figure 7. Relationship between PDF and CDF function [9].

Cumulative distribution function value at given point represents the probability that the result is the xi or less. CDF function is integral of PDF function [9].

$$CDF(x_i) = \sum_{k=1}^{j} N(x_k) \quad (4)$$

CDF is a cumulative distribution function
X_j is the result
$N(x_k)$ is round k probability density function

Formula 5 defines the Empirical Cumulative Density Function (ECDF), which is used to estimate CDF from data samples. ECDF is a step function, which is composed from a group of data samples.

$$EDCF(z) \triangleq \frac{1}{n} \sum_{i=1}^{n} 1\,(Zn \leq \mathrm{Z}) \quad (5)$$

ECDF is a empirical cumulative density function
1(.) is a indicator function
Z is a sample
n is a amount of samples

Indicator function in formula 5 equals one if the indicator function input is true. So the output of the ECDF function is sum of indicator functions which are scaled to one [14].

In order to produce ECDF function x axis from the measured sample group, the group must be sorted from shortest delay to longest delay. Since the first sample of Y axis value is 1/n, whole Y axis consist of evenly distributed numbers from values 1/n to 1.

2.1.6 Radio Link Measurements

In order to isolate problems in radio channel the radio characteristics should be observed. Signal to noise (SNR) ratio is a common radio channel metric which describes a ratio of wanted signal and unwanted signals [15]. Wanted signal is for example the Lora signal that will be detected and common unwanted signal is noise. When the wanted

signal starts to degrade, in other words approach the noise level. Then the signal to noise ratio starts to degrade also. At certain power level, the receiver detector is not able to detect the symbols and bit error ratio is increased. This point is called the receiver sensitivity.

These power levels are called Received Signal Strength Indicator (RSSI) values [15]. As a conclusion if a transmitter power level is constant and a radio link distance is increased the received power level starts to decrease and the signal is approaching the receiver noise floor. This means that RSSI and SNR values are decreasing while the link distance is increased.

It is also possible that the RSSI level is constant and the SNR value is decreased. In this case some interference is the dominating unwanted signal power. So with RSSI and SNR the link characteristics can be defined and the source of the link quality issues can be isolated. The source can be also an insufficient link margin, which can be observed as low RSSI and low SNR value.

3 Fundamentals of LoRaWAN Specification

According to the LoRaWAN specification, the LoRaWAN system is a star of stars type of a network where devices are spreading the communication to different sub channels at different data speeds. LoRaWAN network protocol is optimized for battery-powered systems. Network end devices are able to operate in mobile or fixed-point applications [16 pp.4].

3.1 Air Interface

The core feature of LoRaWAN technology is a long range radio link. According to a Semtech LoraWaN modem datasheet in Figure 8, LoRa Radio Frequency (RF) interface is based on spread spectrum chirp modulation, which is known for great receiver sensitivity figures [17, pp19].

RFS_L125	RF sensitivity, Long-Range Mode, highest LNA gain, LNA boost, 125 kHz bandwidth using split Rx/Tx path	SF = 6	-	-121	-	dBm
		SF = 7	-	-124	-	dBm
		SF = 8	-	-127	-	dBm
		SF = 9	-	-130	-	dBm
		SF = 10	-	-133	-	dBm
		SF = 11	-	-135	-	dBm
		SF = 12	-	-137	-	dBm

Figure 8. LoRaWAN sensitivity figures [16 pp. 19].

The LoRaWAN MAC does not contain a collision avoidance mechanism, the channel access is based on a simple retransmit when a communication fails. So LoRaWan MAC implementation resembles ALOHA protocol but in addition it includes an adaptive data rate scheme[18].

Additionally there are no repeaters or MESH functionality in the LoRaWAN networks [18]. Therefore, the protocol overhead is minimal and this improves the network throughput. On the contrary, where the overhead increases, there is also more throughput available because the transmission from a gateway to a server is TCP/IP based communication and these devices are not battery powered devices.

There are several reasons why the LoRaWAN technology suits well for many IoT applications. The LoRaWAN Media Access Control (MAC) is fully bidirectional and it supports

multicast messaging [16]. Multicast messaging is used to handle the mass messaging efficiently. For example without multicasting software update process over thousands of network components can be time consuming process in low throughput networks.

LoRaWAN gateways are also able to receive up to 9 end devices simultaneously. This is based on transmission sub bands orthogonality and the quasi-orthogonality of spreading factors [18].

As Formula 6 below defines the LoRaWAN MAC supports adaptive data rate scheme, which means that in short link distances the and nodes are able to use higher data speeds than in long link distances [18 pp.6]. LoRaWAN is also able to increase the interference tolerance in the same way. These features are based on the manipulation of a spreading factor. Additionally LoRaWAN is using coding to increase a link distance and an interference tolerance [18 pp.6]. Coding has the same side effect as the spreading factor manipulation. When a link distance and an interference tolerance increases, the data rate is going to decrease.

$$Rb = SF * \frac{BW}{2^{SF}} * CR \quad (6)$$

Rb is a bitrate
SF is a spreading factor
BW is a bandwidth
CR is a coding ratio

LoRaWAN includes some technical constrains due to radio band restrictions. One key limiting factor is that it is using Industrial, Scientific and Medical (ISM) radio bands with 1% duty cycle limitation [19 pp.8]. According to Formula 7, LoRaWAN transmitter is able to transmit 36 seconds in 1 hour period [20 pp. 36].

$$Ts = Ta\ (\frac{1}{d} - 1) \quad (7)$$

Ts is a minimum off period time

Ta is a time on air

D is duty cycle

However, LoRaWAN specification defines that there must be at least three sub-channels in order to fulfill specification requirements. This duty-cycle restriction concerns each sub-channel and each channel at time. Radio regulations in ISM radio bands defines that if frequency hopping is used, the sub-channel hopping sequence has to fulfill duty cycle limit [21]. As in simplified Figure 9, this means that single sub-channel duty cycle increases. This is not so simple situation in practice because duty cycle changes according to air time which is depended on radio link characteristic.

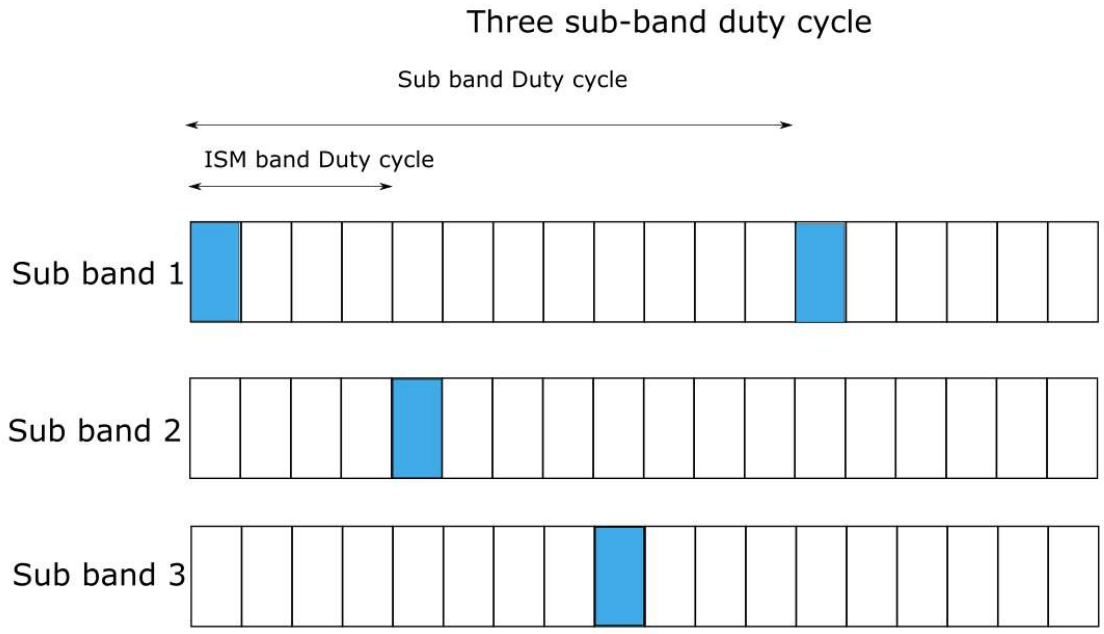

Figure 9. Duty cycle in three sub-band system.

Figure 10 shows time on air values with different MAC payload sizes, which are based on Formula 5. The coding factor was 4/5 in this case. The higher spreading factors and packet sizes increases the time on air and therefore the off period time also increased [20 pp. 36]. This means that when long link distances are desired the data rate decreases due to Formula 5. Additionally the maximum received packet count per hour is decreased due required off period time.

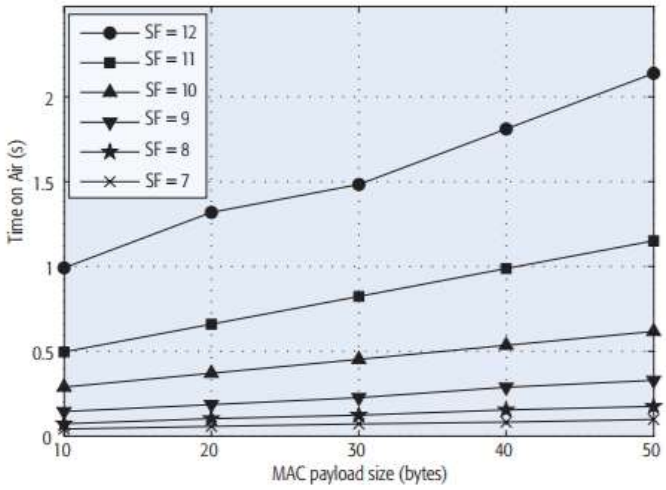

Figure 10. Time on air with different spreading factors [19 pp. 36].

According to Adelelanto et al studies the LoRaWAN network performance drops down quickly when load is increased due to a simple channel access scheme (see Figure 11). The packets collide when transmissions are on the same channel and the same spreading factor is used [20 pp.36].

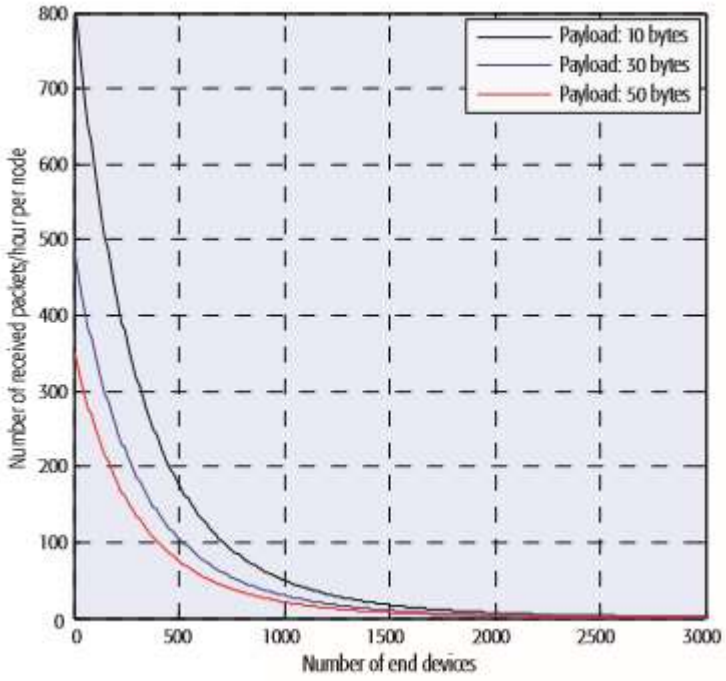

Figure 11. LoRaWAN network performance when packet are colliding [10 pp.36].

As illustrated in figure 12, in three sub channel low throughput networks the duty cycle is limiting the maximum throughput and on the contrary the collisions limits the maximum

throughput until the duty cycle start to affect [20 pp.37]. Finally, in all cases when end device count is increased the throughput starts to fall. N is a number of end devices in the network.

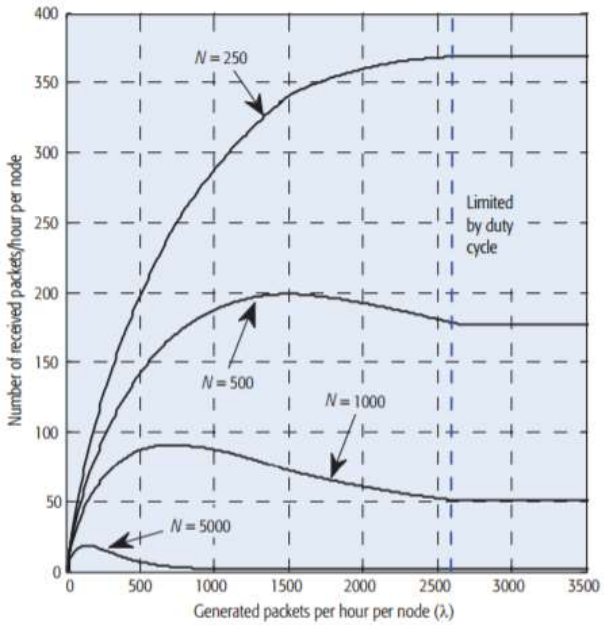

Figure 12. Number of received packets as a function of generated packets [20 pp.37].

Previous calculations were done to demonstrate the LoRaWAN air interface behavior. Additionally the mac layer efficiency does affect to air time. In Figure 13 the overall air time consist of preamble symbols, header, CRC, payload and payload CRC [22].

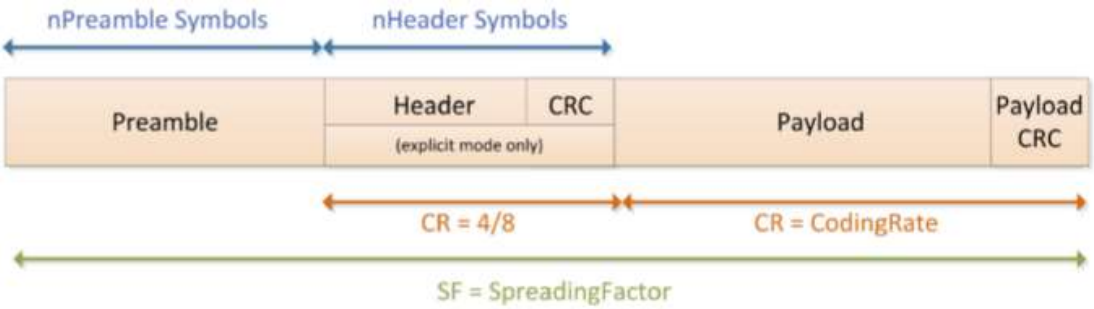

Figure 13. LoRaWAN packet format structure [22].

As a conclusion LoRaWAN has a few weaknesses that degrades the network performance when the network load is increased. LoRaWAN is operating in ISM radio bands so other LoRaWAN application end devices can affect to system performance. Due to duty cycling LoRaWAN is not suitable in real time systems and also in applications where low network jitter is required.

In following chapters the LoRaWAN network architecture and components and their interfaces are explained.

3.2 Network Architecture

As can be seen in figure 14, core of the LoraWAN network architecture is the Network server. In order to achieve star-to-star topology, there can be many gateways connected to network server. Additionally End devices are able to connect to several gateways [16].

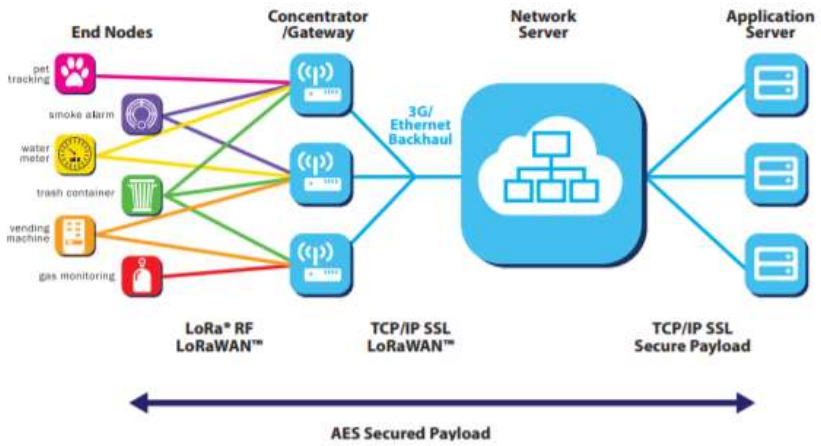

Figure 14. LoRaWAN architecture [3 pp.8].

First and the closest network component to client is the Application server, which is presented in next chapter.

3.2.1 Application Servers

Application servers are connected to a network server, which are analysing the data from thousands of network nodes. A typical example of an application server is an IoT ticket solution [23]. This platform provides a web UI, a remote control, data reporting, analytics and monitoring of assets. Figure 15 shows the access to applications servers are provided by REST Application Program Interfaces (APIs).

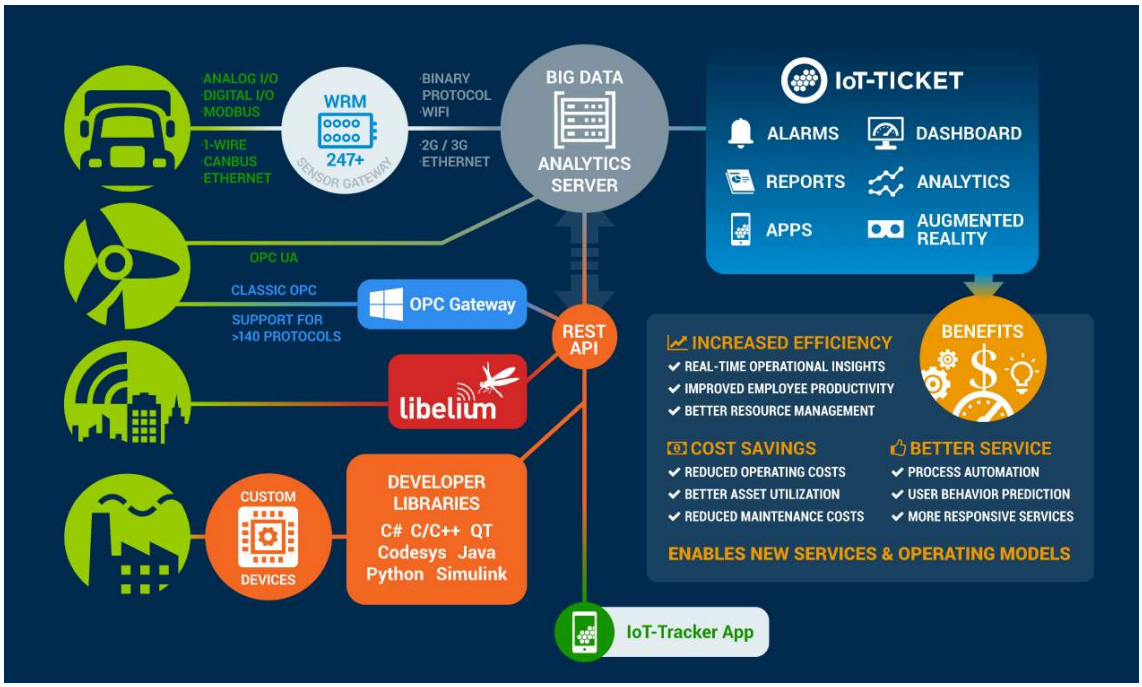

Figure 15. Wapice IoT ticket solution [23].

The REST is a simple and web-friendly interface for designing web services. The Rest API consist of simple HTTP commands such as POST, GET and PUT [24]. The REST is also utilizing directly HTTP functionalities and possibilities and does not add more layers on top of the HTTP. However, even the REST is a simple, lightweight and easy to apply, it might not be able to provide adequate service for system, which has complex functionality [24 pp.100].

3.2.2 Network Server

Network server is responsible for network intelligence and the LoRaWAN transmission is executed via TCP/IP backhaul interface. It regulates the end device spreading factor to adapt the data rate according to radio link circumstances. In other words it maintains the adaptive data rate scheme [16 pp.3].

The network server also filters duplicate packets and sends acknowledge messages to gateways. The network server also chooses the best gateway, therefore there is no need for handover procedure. The network server also sends the messages to a specific application server. Additionally the end device to end device communication is typically handled via network server. Finally the network server is also responsible for the end device authentication and security [2 pp.16].

3.2.3 LoRaWAN Gateway

In Figure 16, LoRaWAN gateway receives and transmits packets via wireless link from End devices. These packets are forwarded to LoRaWAN server via TCP/IP backhaul interface.

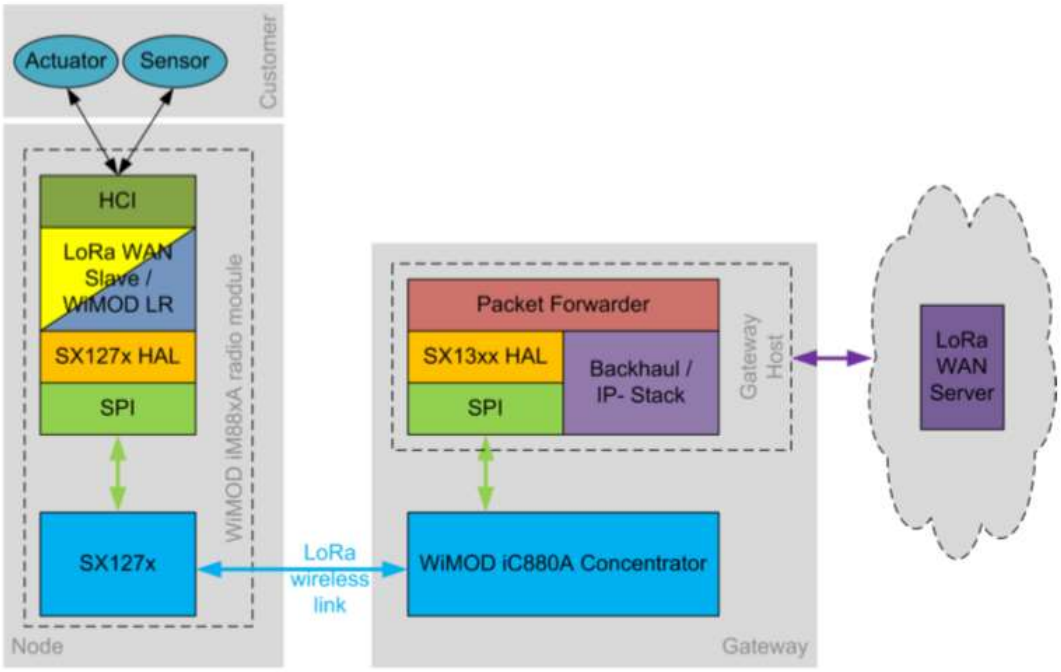

Figure 16. Gateway structure [25].

LoRaWAN gateway handles the timing of the downlink messages through RF interface. Because end devices do not contain destination addresses available, the gateway acts as a relay between the end devices and a network server. Several gateways can receive a same message from an end device [16]. This feature can be used to extend the network range and a redundancy is built automatically in to a LoRaWAN network. Also this feature ensures that hand-over is not needed [16].

3.2.4 End Devices

End devices such as sensors can be set to three different operation modes. A maximum latency means a minimum power consumption and vice versa. The LoRaWAN MAC is

utilizing a device sleep time to enhance the power efficiency and it is also possible to change the operation class during the device operation [2 pp.19].

Figure 17 shows that the longest battery life is achieved with a class A operation mode. This is archieved with exact end device receive time intervals, between the time intervals the device is in a sleep mode [26]. After the data is send to gateway, class A device listens response during two downlink receive windows. Typical offset time between these windows is 1 second.

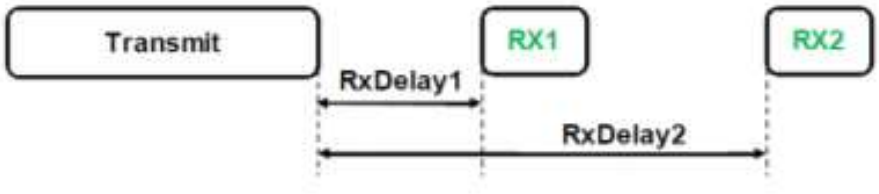

Figure 17. Class A operation mode [26 pp.2].

In Figure 18, Class B devices are actuators which are connected to network in Class C operation mode. Then device decides whether it continues on C mode or changes the operation mode to class B. Class B device opens extra receiving window in order to synchronize with gateways beacon messages in regular time interval [26].

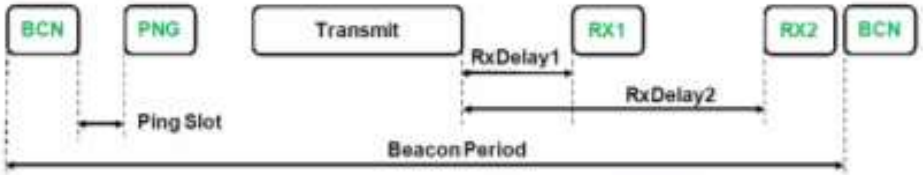

Figure 18. Class B operation mode beacon period [26 pp.2].

Class C has the lowest latency because the end device keeps the receiver typically on except during transmission [26].

LoRaWAN specification allows end devices to send data at any time on any available channel when following rules are used. The transmission channel is changed in a pseudo-random fashion. This will cause a frequency diversion, which makes the system more tolerant to interferences. Finally end devices have to respect the duty cycle and maximum transmit time requirements due to local radio regulations [16 pp. 6].

3.3 Security and Network Access

Security is one of the most critical issues in IoT applications. Without proper security, the IoT applications are providing ends nodes as access points to cyber attacks. As illustrated in Figure 19, the LoRaWAN security is divided in two different layers according to LoRaWAN network structure and each layer has its own cyphering and singing method [13].

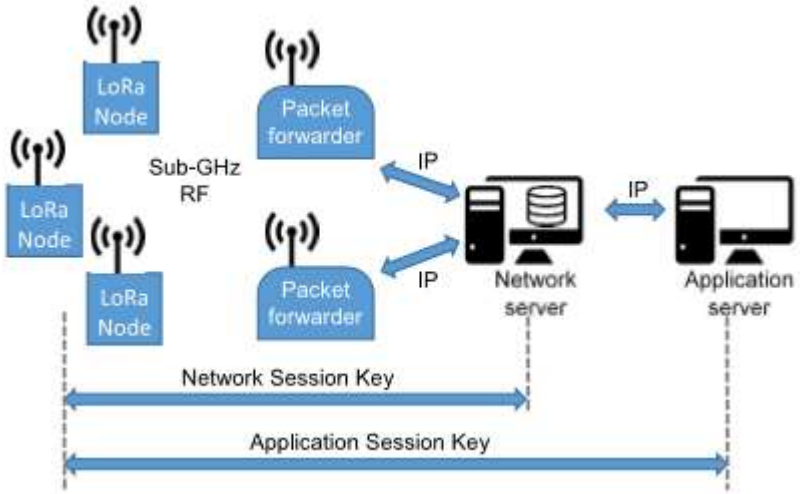

Figure 19. Security layers in LoRaWAN architecture [13].

The end device authentication is ensured by the network layer and on the other hand the application authentication is provided by the application layer. So both layers consist of different keys to distinguish layers from each other. These keys are called application and network session keys [13]. This method is useful if the network hardware is provided by a different company than the company which owns the application.

There are two different activation methods available in the LoRaWAN system. One is to use the Over the Air Activation (OTAA) method which is based on unique network identifier and over the air message hand shaking. Other one is the activation by a person that uses the activation keys which are stored in LoRaWAN devices already in a production process [13].

3.3.1 Adaptive Data Rate

LoRaWAN end devices and gateways are able to use any of the supported spreading factor's. So the static links between the gateway and end devices are optimized in order to achieve the fastest possible data rate. The network server controls the data rate via appropriate MAC commands [16].

As Figure 20 illustrates the fastest data rate is achieved when the link distance is short and vice versa.

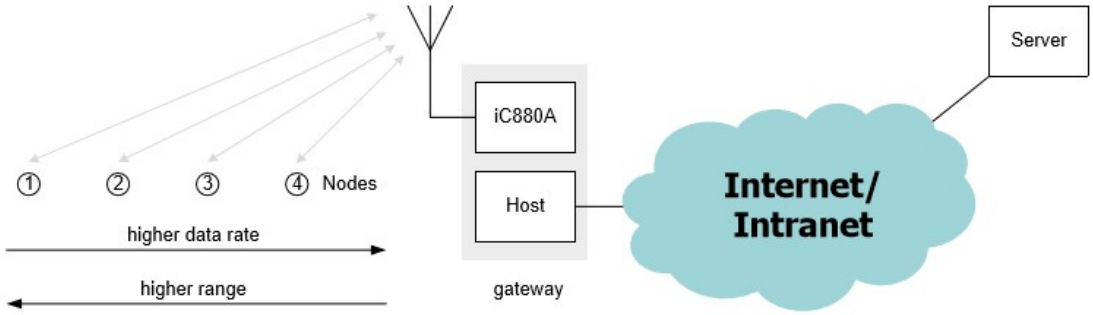

Figure 20. Adaptive data rate [26].

When the optimum data rate is set, the status of the link between the end device and gateway has to be verified. This is executed by the end device. When the end device has transmitted its data, the delay time of the gateway acknowledge message is measured. If a certain delay period has passed, the end device reduces the data rate by one step [16].

On the other hand, it is not possible to adjust the data rate when radio parameters such as attenuation varies rapidly. In this case the network is not able to control the data rate, so the device application layer should control it with a predefined set of data rates [16].

It is also possible to set the Adaptive Data Rate (ADR) off by setting ADR bit from control header off. However, the LoRaWAN specification recommends to use this method in order to increase the battery lifetime. Finally if the end device is able to detect its mobility, it can request the network to set the adaptive data rate on [16].

4 Solution Model for Evaluating LPWA Technologies

This solution model provides a tool for evaluating LPWA technology. The core principle is simple, send the time stamped test data to network and calculate QoS metrics, which are based on test message header information. Solution model greatest challenges are on measurement accuracy and reliability. Evaluation platform performance should be better than the communication system under investigation. Also the platform scalability is one of the core features.

Reference data is calculated to analyze the communication system performance. This calculation is based on formulas that are presented in next chapter, LoRaWAN data packet structure and the end device operation mode.

4.1 Air Interface Reference Delay Calculation

Fundamental time unit in calculations is the symbol duration [22]. As the formula 8 illustrates, this is calculated with the transmission spreading factor and the bandwidth.

$$Tsym = \frac{2SF}{BW} \tag{8}$$

Tsym is a symbol duration
SF is a spreading factor
BW is a channel bandwidth

First common parameter to all transmissions is the pre-ample section. It consist of pre-programmed pre-ample symbols. From these symbols (formula 9) the sequential pre-ample section is generated [22].

$$Tpreample = \left(n_{preample} + 4.25\right)Tsym \tag{9}$$

Tpreample is a preample duration
n_preample is a preamble length in symbols
Tsym is a symbol duration

As in formula 10 [22], in order to calculate the overall air time, the number of payload symbols must be calculated.

$$Payloadsymbolscount = 8 + \max((\text{ceil}(\frac{8PL+4SF+28+16}{4(SF-2\quad)})(CR+4), 0) \tag{10}$$

Payloadsymbolscount is a overall symbol count
PL is a number of payload bytes
SF is a spreading factor
H is a header set boolean type number
DE is the low bitrate optimization Boolean type of number
CR is the coding rate (values 1 to 4)

The payload overall time can be calculated from formula 11 [22].

$$T_{payload} = Payloadsymbolscount * T_{sym} \tag{11}$$

$T_{payload}$ is a payload air time
Payloadsymbolscount is a overall symbol count
T_{sym} is a symbol duration

The overall time on air is the combination on the pre-ample section and the actual payload section. In formula 12, with the symbol duration, the overall time can be calculated [22].

$$T_{packet} = T_{payload} + T_{preamble} \tag{12}$$

T_{packet} is a overall air time
$T_{payload}$ is a payload overall time
$T_{preample}$ is a preample time

In order to calculate overall transmission delay, the minimum off time has to be calculated [22]

$$T_{off} = T_{packet}\ (\frac{1}{d} - 1) \tag{13}$$

Toff is a minimum off period time

Tpacket is a time on air

D is duty cycle

So if the data packet are send sequentially, there has to be minimum off time between the transmissions, otherwise the transmitter rejects the transmission due to duty cycle limitation.

In addition the LoRaWAN protocol requires some air time. This is included in PHYPayload in Figure 21 [16].

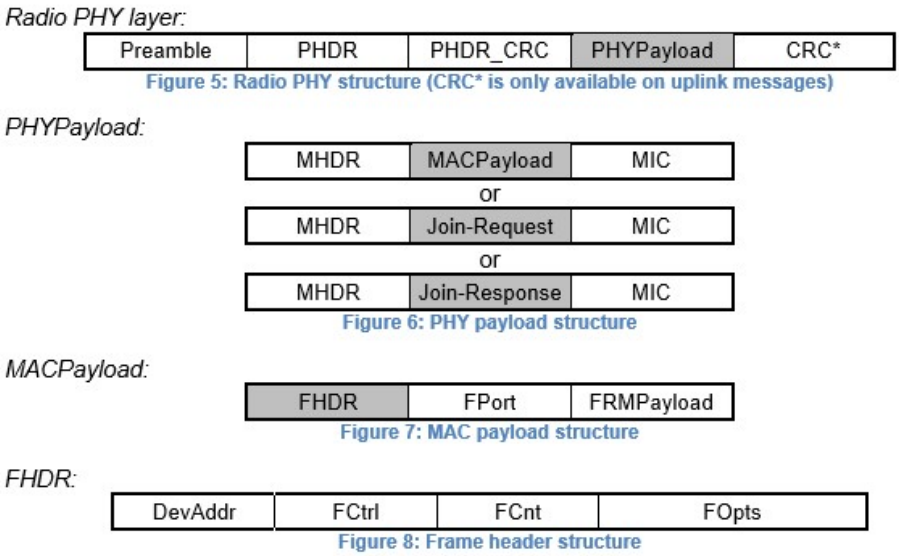

Figure 21. Frame header structure [16].

Additionally, PHYPayload has to be included to application payload byte count in formula 8. PhyPayload byte count is defined in table 2. According to LoRaWAN specification, all other header bytes are constant but Fops header section varies between 0 and 15 bytes [16].

Table 2. Minimum and maximum Phypayload in bytes [16].

PhyPayload	13		
MHDR	**MACpayload**	**MIC**	
1	8	4	
MACpayload			
FHDR	**Fport**	**FRMPayload**	
7	1	0	
FHDR			
Devaddr	**FCtrl**	**FCnt**	**Fops**
4	1	2	0

PhyPayload	28		
MHDR	**MACpayload**	**MIC**	
1	23	4	
MACpayload			
FHDR	**Fport**	**FRMPayload**	
22	1	0	
FHDR			
Devaddr	**FCtrl**	**FCnt**	**Fops**
4	1	2	15

MAC commands are transported in Fopts section. These commands are used between the network server and end device communication. Commands are used to control the receiver timings, data rate, duty cycle and channel settings. Furthermore with these commands the status of end device is send to Network server. This status data consist of battery level and demodulation margin [16].

So the amount of LoRaWAN protocol bytes is difficult to estimate since the Fopts contents vary according LoRaWAN network state. Therefore in this thesis the maximum and minimum amount of Phypayload bytes are used to evaluate air interface delay. The actual air interface transmission delay has to be in between these values.

4.2 LPWA Evaluation Platform

As illustrated in Figure 22, the solution model consist of an evaluation network which is feeding test messages through the LoRaWAN network.

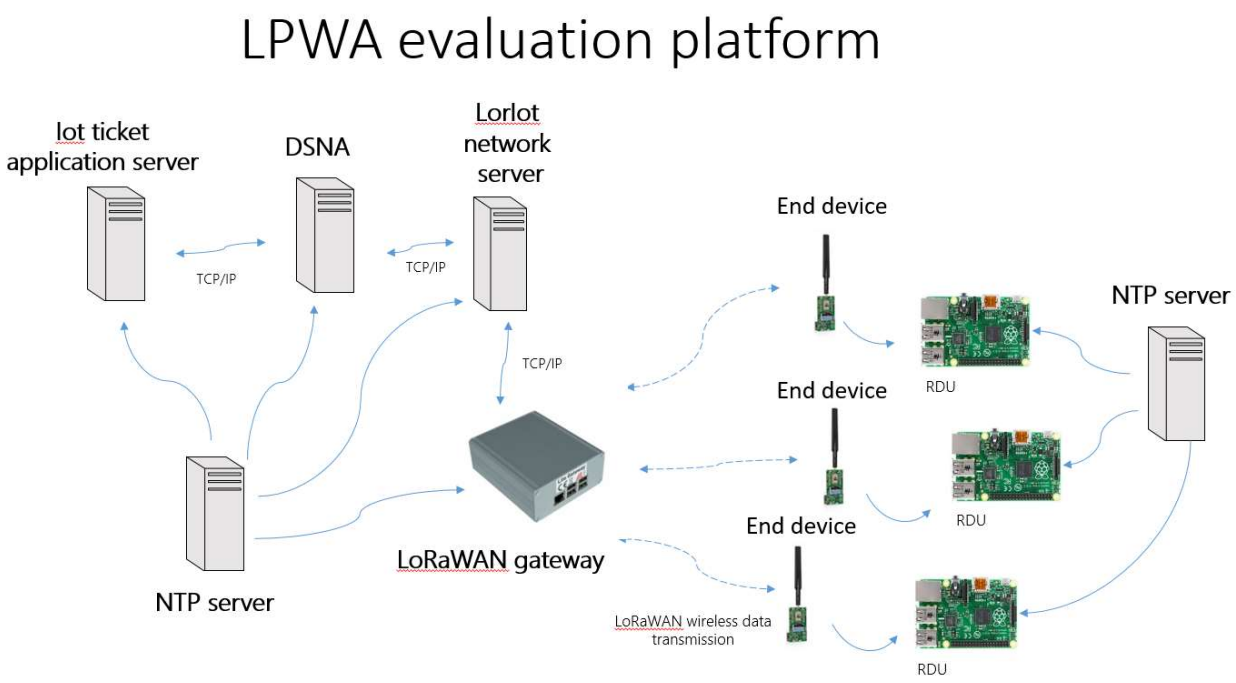

Figure 22. Evaluation platform architecture.

Test messages are generated by Raspberry pi Diagnostic Units (RDU), which are connected to end devices. After test messages has been received, they are parsed and

analysed in Diagnostic Server/Network Adapter (DSNA). DSNA sends the test results to IoT ticket application server which is basically creating a test report of analyzed data. Additionally, a NTP server is connected to all LPWA evaluation platform components in order to achieve clock synchronization.

RDU's hardware platform is the Raspberry pi mini computer with a dedicated Linux operating system distribution. DSNA is a virtual server running on Wapice premices and the Loriot network server which location is somewhere in Germany. A LoRaWAN gateway and end devices are manufacturer's evaluation platforms.

QoS data sources are defined in transmission files which are send to each RDUs. These transmission files consists of transmission interval, the start time of the transmission, test type and actual data. With this information, the test message header section is created which consist of the transmission type, interval and transmission number. According to these parameters the QoS data is calculated in the DSNA diagnostic process.

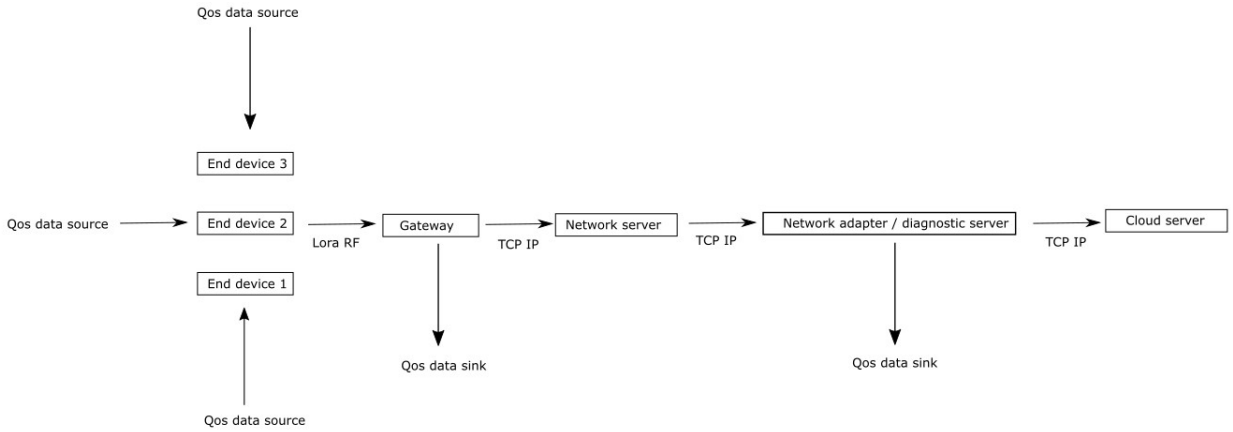

Figure 23. Extended quality of service.

Figure 23 shows the traditional QoS was extended so that the metrics are measured from gateway and DSNA and the radio channel metrics are also available. So more data is available in a situation when a defect is degrading the overall performance to isolate where the actual problem is.

4.2.1 Application Server

Evaluation platform application server is Wapice IoT ticket solution. This server is providing analysis tools for displaying the QoS measurement results. So with an IoT ticket, the evaluation platform itself is acting as a true IoT application.

In order to connect the device to IoT ticket the end devices are defined to IoT ticket account. There is one end device defined in Figure 24.

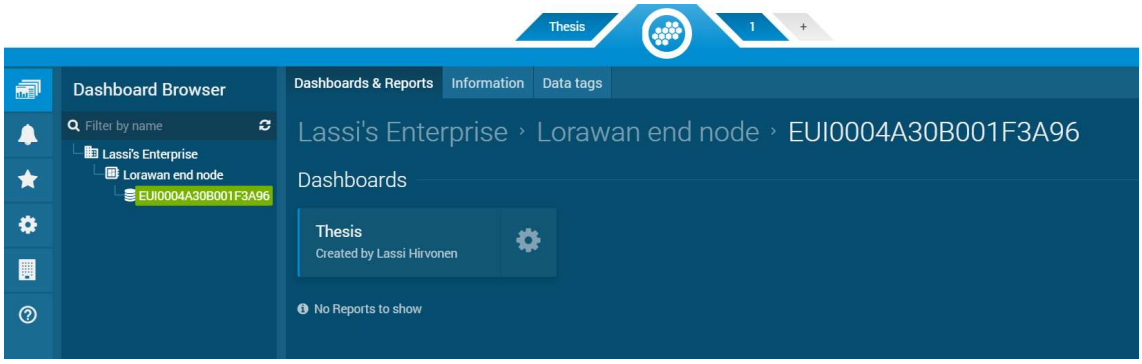

Figure 24. Iot ticket dashboard browser.

Additionally the data nodes in other words data input have to be defined in data tags section. Actually this is executed in network server IoT ticket interface but when it is defined properly those tags can be seen in Figure 25. Network server is defined in chapter 4.2.2.

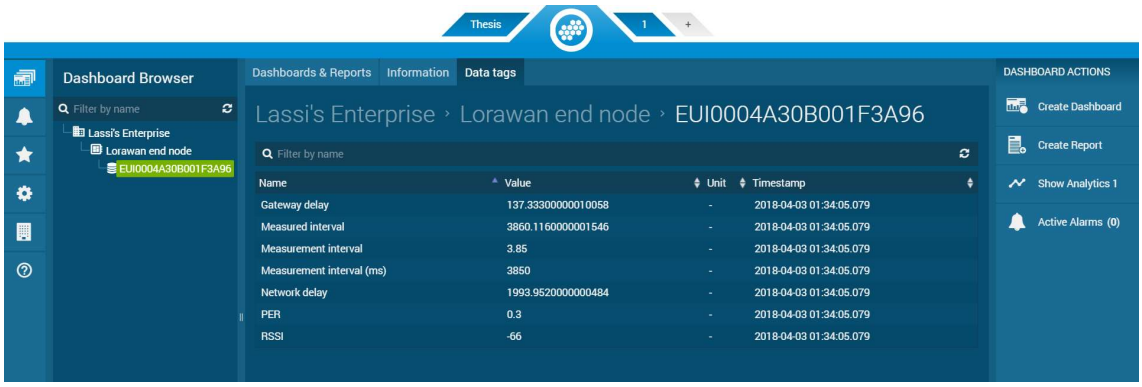

Figure 25. Iot ticket data tags section.

Now the actual measurement can be set by clicking the create dashboard button which opens the interface designer. Figure 26 shows that the graphs are created with UI element's tool. Now data nodes are connected to each graph. Figure 28 presents the finished interface design.

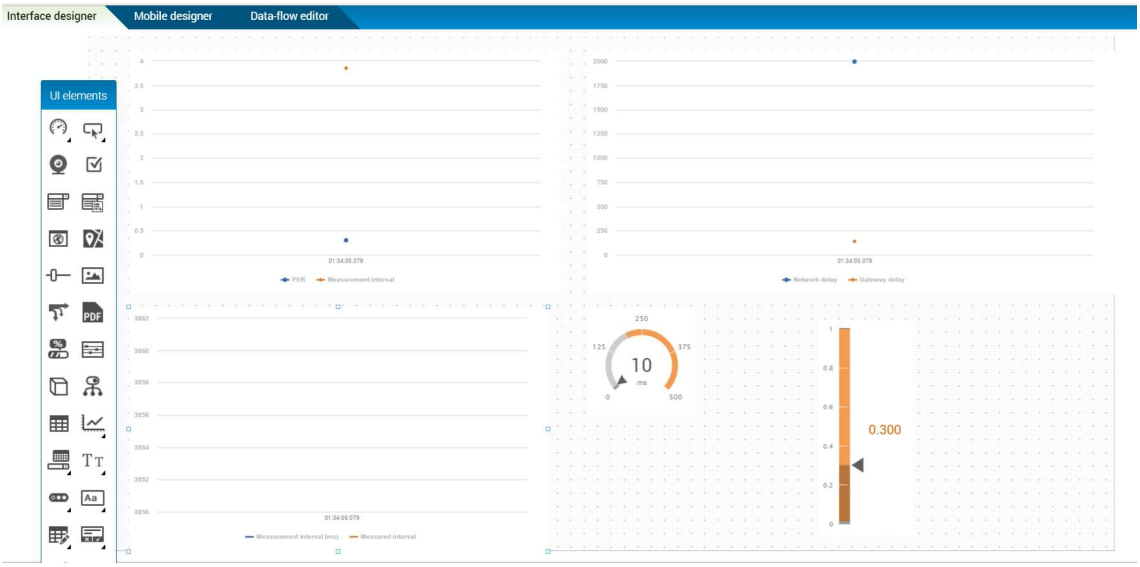

Figure 26. Interface designer.

There was a demo session in a Wapice customer event where the first version of the evaluation platform was introduced. In Figure 27, this session duty cycle and different delays in different network components were presented.

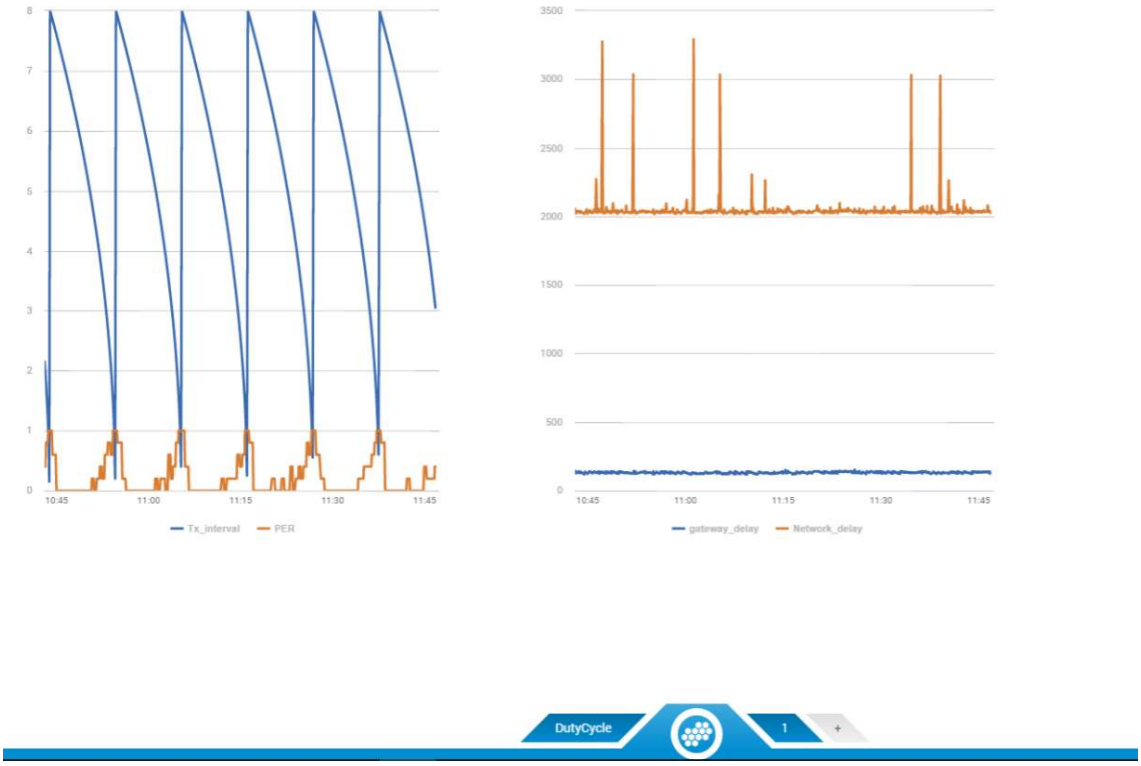

Figure 27. IoT ticket application server in action (customer demo session day).

The network server should be connected directly to the application server but there was a compatibility issue between the Lortio network server and IoT ticket interface. The Json data interface did not match, so in order to connect IoT ticket to the evaluation platform, the network adapter was implemented.

4.2.2 Diagnostic Server / Network Adapter

In Figure 28 the network adapter consist of a HTTP server which is receiving post messages from the Loriot network server. These post messages are parsed, processed and send through the IoT ticket interface. This interface is a python library provided by Wapice ltd which is used to define and send the data nodes through the REST interface. Iot ticket interface is presented in appendix 1.

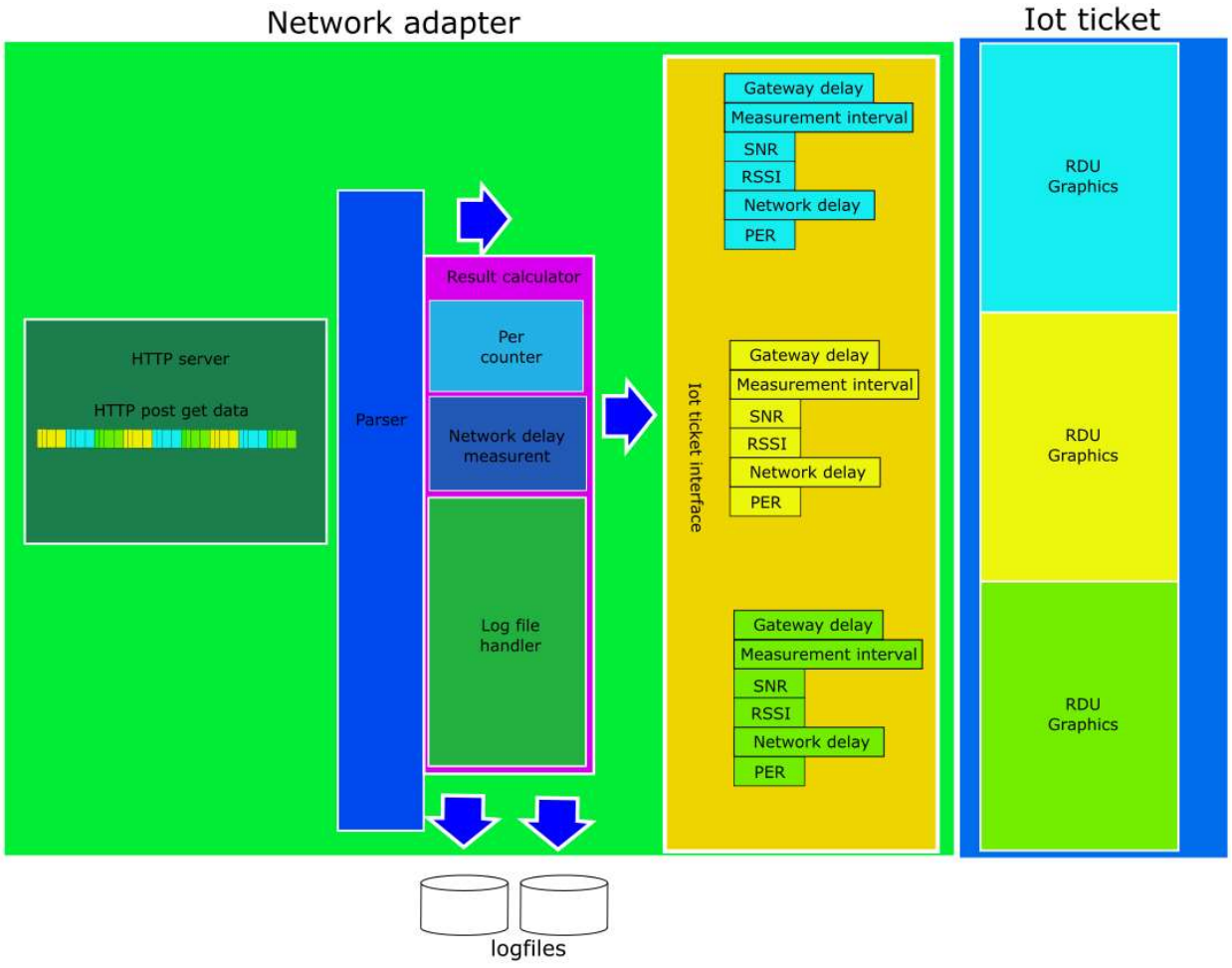

Figure 28. Network adapter data handling.

Data processing in DSNA means transmission delay and Packet Error Ratio (PER) calculation. This delay calculation is based on time stamps, which are created during the transmit process. The calculated transmission delays are also stored to log files. These files are used in ECDF curve calculation. The result calculator calculates also the reference delay values according to chapter 4.1 formulas.

4.2.3 Network Server

Basic function of network server is to manage the network, session keys, security and privacy. Additionally server is providing the LoRaWAN protocol to IP/IPv6 translation [28].

The LPWA evaluation platform is utilizing the Loriot network server shared instance. This instance has basic functionalities to start evaluating LoRaWAN system [28]:

- One free gateway connectivity slot
- One free network application
- Capacity of 10 devices
- Streaming API
- Cloud to cloud output
- Only uplink mode
- OTAA activation
- Join server interface

To connect the gateway to the network server the gateway MAC has to be defined in Loriot user interface. When the gateway is configured to a Loriot mode, the basis of the network is set to start transmission.

4.2.4 LoRaWAN Gateway

Lorawan lite is LoRaWAN evaluation and demonstration platform. It consist of a LoRaWAN concentrator board and a Raspberry Pi (RPi) mini computer. It is preconfigured so that it is easy to connect Loriot or semtech network servers [27]. Rpi firmware is an open source github project which already supports the loriot server connection [29].

According to the github project documentation, the gateway upstream Json data structure is illustraded in listning 1. Upstream data consist of few diagnostic parameters, which are utilized in this thesis. Time, rssi data and lsnr are the Json object parameters that are extracted and used to analyze the LoraWAN transmission quality [29].

```
{"rxpk":[
            {
                        "time":"2013-03-31T16:21:17.528002Z",
                        "tmst":3512348611,
                        "chan":2,
                        "rfch":0,
                        "freq":866.349812,
                        "stat":1,
                        "modu":"LORA",
                        "datr":"SF7BW125",
                        "codr":"4/6",
                        "rssi":-35,
                        "lsnr":5.1,
                        "size":32,
                        "data":"-DS4CGaDCdG+48eJNM3VaizDpsR71P"
            }
]}
```

Listning 1. Upstream JSON data structure.

In Figure 29, the Rpi computer is connected to an i880A concentrator board which is basically a multichannel transmitter/receiver module.

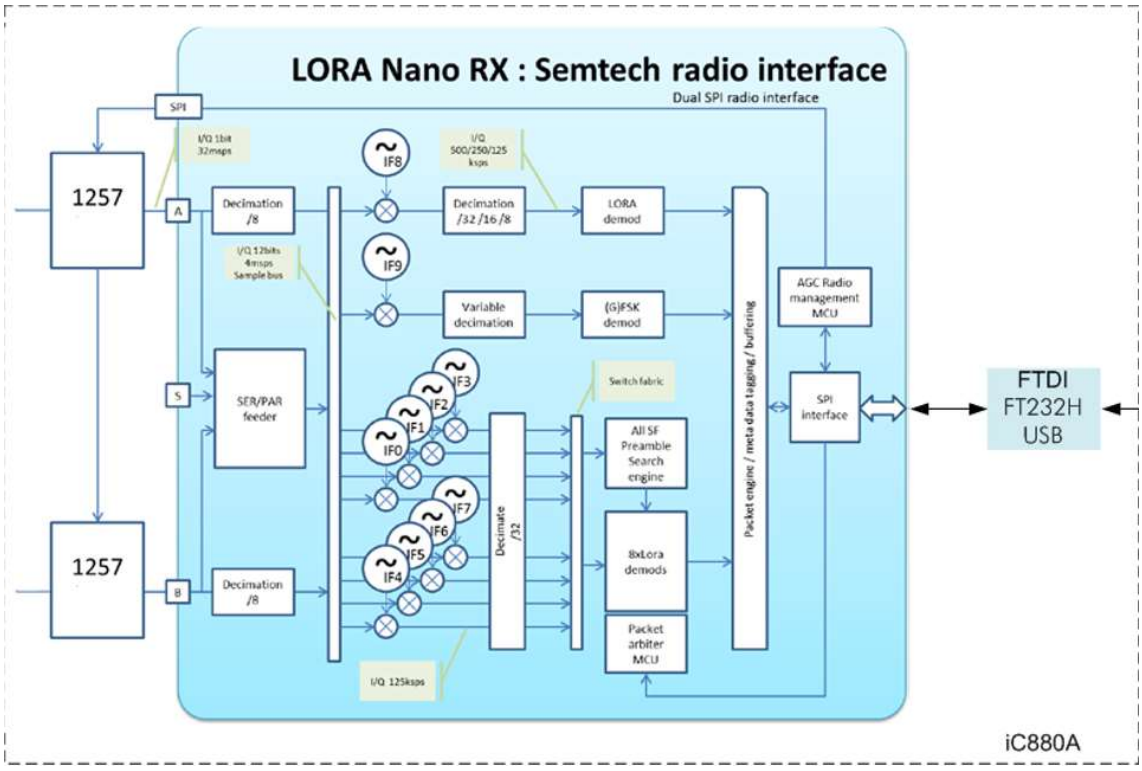

Figure 29. i880A concentrator board RF section [27].

Due to the i880A receiver architecture, this module is able to receive 8 LoRaWAN packets simultaneously due to orthogonality of spreading factors [27]. This feature enables several benefits in LoraWAN radio interface:

- Frequency can be changed with each transmission in a random pattern with improves the system interference tolerance and radio channel diversity
- All data is demodulated in parallel, so end devices are able to adapt their data rate and there is no need to maintain data rate tables.
- The capacity of the air interface can be increased due to orthogonal spreading factors.

Finally since the receiver sensitivity is down to -138 dBm the radio link range is high. So a star topology can be used in order to avoid complex network layers, wireless routers and additional network protocol traffic [27]. Eventually this saves the battery lifetime.

4.2.5 End Device – RDU Connection

DM164138 is an evaluation board for evaluating a low power RN2483 transceiver module [30]. This module is marked as number 6 in Figure 30 evaluation board layout.

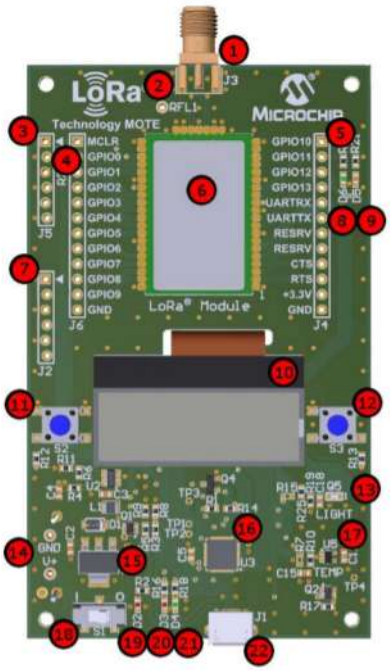

Figure 30. DM164138 evaluation board layout [29].

The RN2483 complies the LoRaWAN specification class A requirements. Figure 31 shows that external MCU is required for a complete sensor application [30].

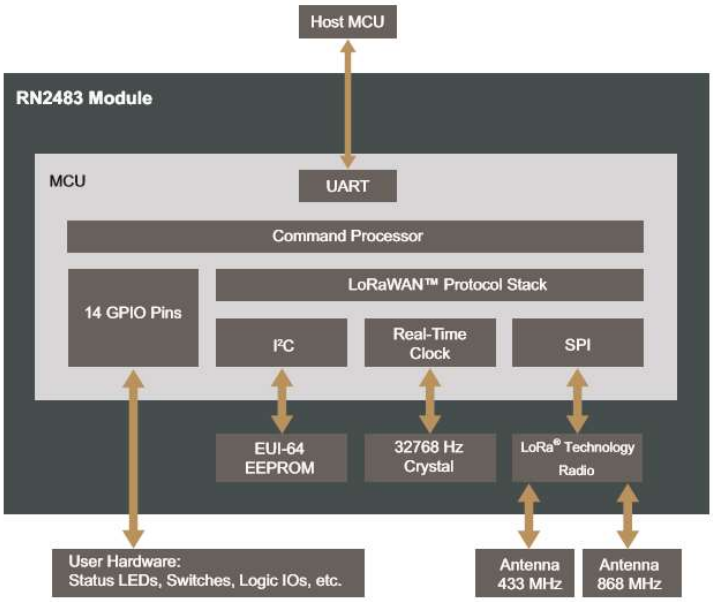

Figure 31. RN2483 module and its interfaces.

In Figure 32, the evaluation board has an USB interface for controlling the module and a digital output for debugging the device [30]. So controlling the evaluation board is executed via USB interface. From the RPi side the end device is seen as a serial port. After the serial port is initialized, the end device is available for actual end device initialization.

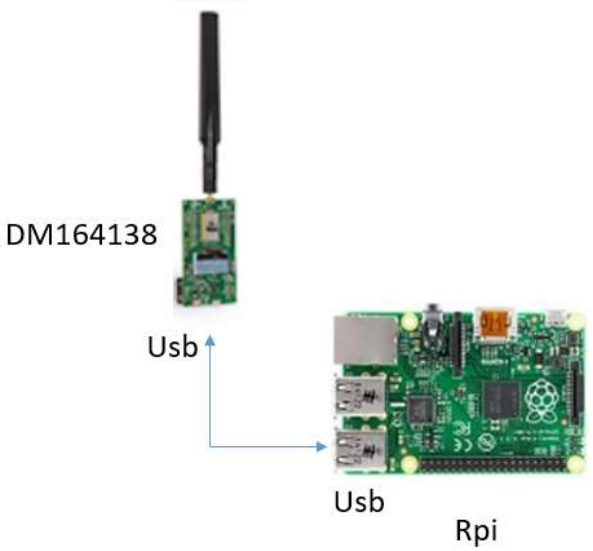

Figure 32. Diagnostic data node and LoRaWAN end device interface.

In this thesis work the network connection was carried out with an OTAA method. The OTAA method requires three different keys to connect to the network.

Below are the minimum OTAA interface commands [32]:

- mac set deveui 0123456789ABCDEF
- mac set appeui 0123456789ABCDEF
- mac set appkey 0123456789ABCDEF0123456789ABCDEF
- mac join otaa

After initialization, data can be transmitted with following instruction set [32]:

- mac tx data portnumber

4.3 Diagnostic Software

Diagnostic software was written with Python language, which is running in all RDU's and in DSNA. First version did not include the user interface. The system is controlled via

configuration files. All functionalities are triggered via files so whenever transmission or configuration file exists in the file system, the execution is started. If the user interface is implemented later on, the system is handled via these files.

First the transceiver examines the transmitter Json file and starts to parse transmit parameters. As in illustrated in Listning 2, these test messages are located in transmission Json file as a Json object

```
self.data['transmit_parameters'].append({
                'start_time': 'none', # seconds.milliseconds from EPOC
                'send_interval_milliseconds': 8000,
                'send_count': 10000,
                'send_forever': 'false',
                'status': 'waitin_to_start',
                'interval_decrement_milliseconds': 100,
                'data_content': 'from_diagnostic_frame'
            })
```

Listing 2. Transmit parameters JSON data structure.

Second the current time and start of the of this measurement is observed with 1 millisecond interval. As in figure 33 illustrated, When the start time has come, the measurement thread is generated. This transmit thread is generating test messages according to this Json information.

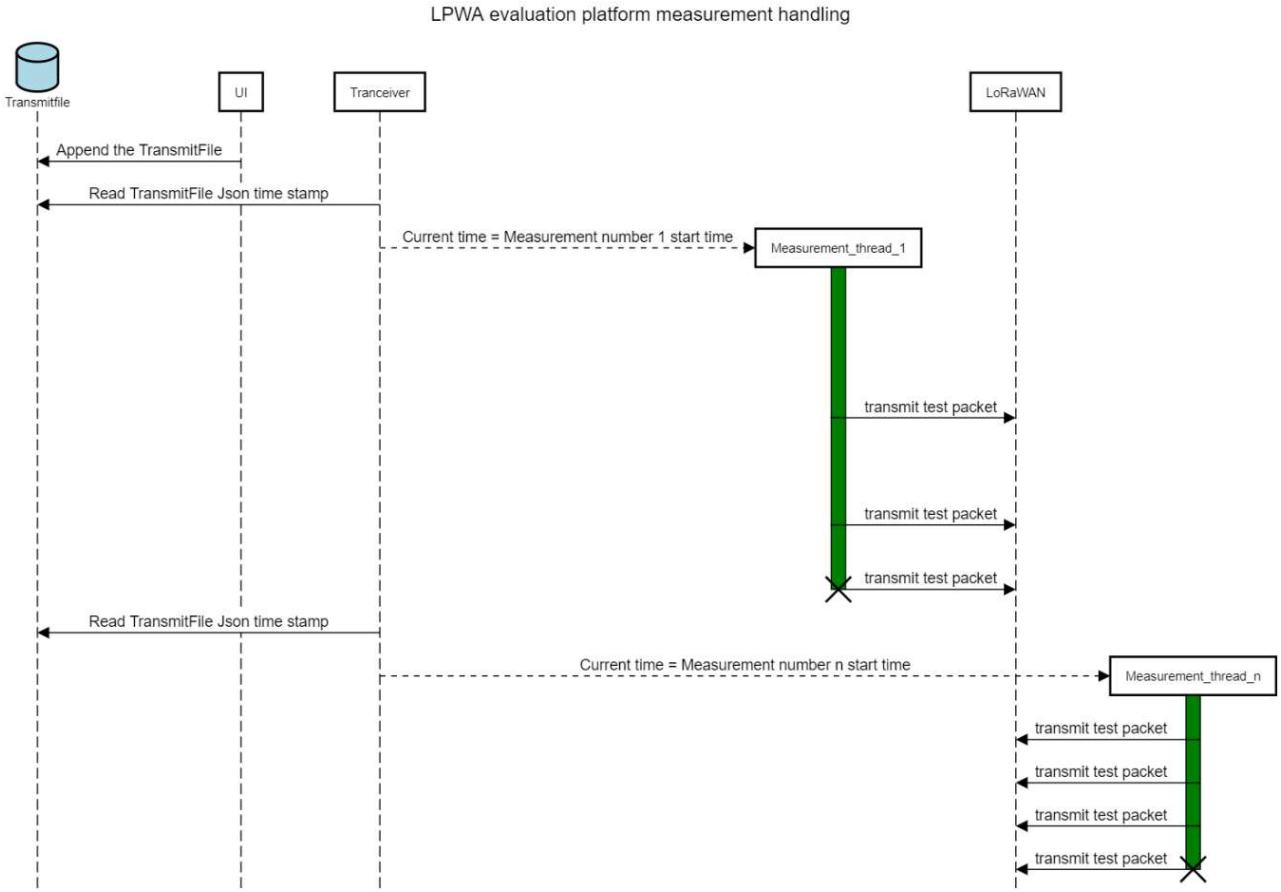

Figure 33. Test transmission functionality.

Third the test message's is generated by transmission thread. Transmission interval, Time stamp and payload is defined in this message header section and the content depends on transmission Json file. Table 3 header is parsed and analysed later on in a network adapter.

Table 3. Test message header structure.

Transmission interval	**Time stamp**	**Payload**
Current transmission sleep time	Time stamp when test message is send from RDU in milliseconds	Transmission number or user defined data

Transmit intervals can be defined as fixed intervals or they can be decreased by a decrement factor.

4.3.1 Diagnostic Software Architecture

Diagnostic software architecture is presented in Figure 34.

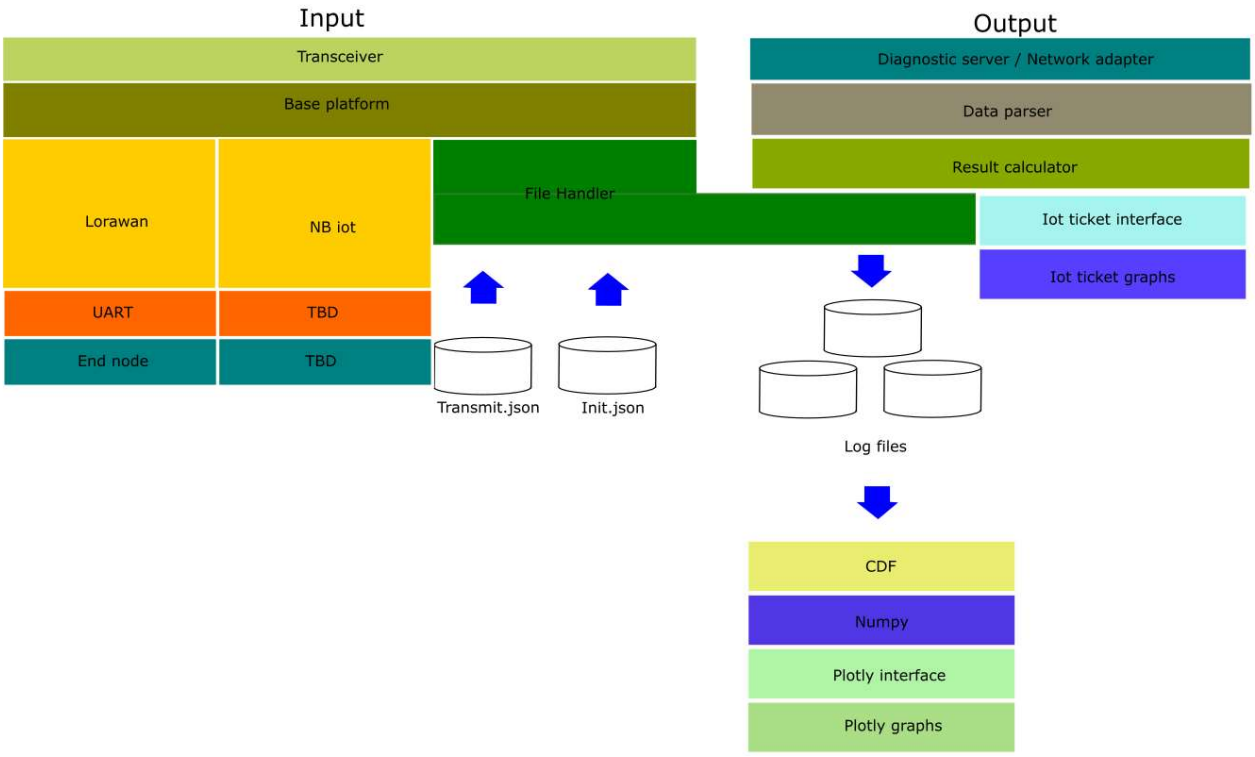

Figure 34. LPWA evaluation system software architecture.

Platform initialization data is defined in an initialization file. This file consist of an end device instruction set and initialization data. For example in a LoRaWAN case, a LoRaWAN end device instance is created according to initialization file content's.

Transceiver starts the transmit process according to the transmit file. The transmission result is received in the network adapter. After the received data is parsed, the data is analysed in the result calculator. Then results are send via the IoT ticket rest interface. The IoT ticket is presenting the short time frame result graphs.

The long term measurement result are available in log files. With these files, statistics are calculated in cdf module which is utilizing Python numpy library in order to calculate CDF curves. Finally, long term time measurements results are plotted with plotly which is a Python graphing library.

When a new LPWA technique is added to the evaluation system, a new platform class has to be created. Also the result calculator and data parser classes must be updated and the system under test output should be connected to the diagnostic server. All other classes should be intact.

4.3.2 Result Calculator

The core of the evaluation platform is the result calculator. The input data contains delays and reference data which is parsed from diagnostic data. This data is used in result calculator.

```
input = int(delay), str_time, int(counter), float(transmission), \
                integer_datarate_values[0], integer_datarate_vaues[1],
           integer_datarate_values[2],payload_lengt_in_bytes

if end_node == self.dev1:
           output = self.dev1_result_calculator.calc_result(input)
if end_node == self.dev2:
            output = self.dev2_result_calculator.calc_result(input)
if end_node == self.dev3:
            output = self.dev3_result_calculator.calc_result(input)

return output
```

Listning 3. Result calculator input data.

So, when the transmission setup is changed in listing 3, the result calculator is adapted according to those changed values.

The output data in listing 4 contains the result of the calculation and it is send to the Iot ticket interface.

```
self.data={
            'per': self.per_counter,
            'gatewaydelay': self.gateway_delay,
            'networkdelay': self.network_delay,
            'measinterval': 0,
            'interval':0,
            'interval_ms': 0,
            'calculated_delay':0,
```

```
            'calculated_minimum_off_period':0,
            'calculated_minimum_off_period_from_measured_delay': 0
        }
```

Listning 4. Output data structure.

Actual result calculator method is presented on appendix 2. First the network delays and reference values are calculated. Second the long term measurement data is stored in log files and finally the PER values are calculated.

PER value calculations is based on Cyclic Redudancy Check (CRC) calculation and accoding to Listning 5, on case of CRC error the packet is not send from gateway to Network server [29].

```
"gateway_conf": {
        "gateway_ID": "AA555A0000000000",
           /* change with default server address/ports, or overwrite
           in local_conf.json */
        "server_address": "localhost",
        "serv_port_up": 1680,
        "serv_port_down": 1680,
        /* adjust the following parameters for your network */
        "keepalive_interval": 10,
        "stat_interval": 30,
        "push_timeout_ms": 100,
        /* forward only valid packets */
        "forward_crc_valid": true,
        "forward_crc_error": false,
        "forward_crc_disabled": false
    }
```

Listning 5. Gateway CRC configuration [29].

So when the packet is not send due to CRC error, discontinuation on transmit packet counter is detected and this discontinuation is interpreted as missed packet.

Minimum and maximum reference value calculations are listed in Appendix 3. These calculations are based on chapter 4.1 formulas and LoRaWAN protocol overall minimum and maximum byte count.

4.3.3 Plotting ECDF Curves

Command line CDF curve implementation is presented on appendix 4. First the graph object is created. Second, the file names and plot names are read from input and finally these input data's are fed to graph object. Graph class is presented on appendix 5. It is utilizing the numpy and plotly Python libra-ries.

First, the ECDF curves are calculated with Numpy library. Numpy is a scientific python library which provides among others the statistical tools for analysing data [33]. The empirical discrete data is sorted to x axis and Y axis consist of evenly spaced sequence of step function values. These ECDF outputs is used as a graph input.

Second, the graphs are generated with Plotly library methods. Plotly is an open source tool for composing, editing, and sharing data via the Internet interface [34]. When the plotly account is created, it is possible to create plotly object with plotly.tools.set_credentials_file method. Now the connection should be available and the graph figures can be defined with scatterm, make_subplots and append_trace methods. Finally the generated figure is send to cloud and it is available on plotly web page.

5 Evaluation Platform Performance

The evaluation platform is not allowed to degrade the measurement results. In terms of timing accuracy, the <1ms accuracy is enough for toughest requirement by industrial applications. As illustraded in Table 4, all evaluation platform measurements were executed with the same air interface setup in order to simplify the performance measurements.

Table 4. Air interface configuration.

Data retry	**Number of preample symbols**	**Spreading factor**	**Code rate**	**Low bitrate optimization**	**Header**	**Bandwidth**
Off	16	7	4/5	off	on	125 kHz

So the LoRaWAN network is not set to its optimal settings in terms of packet loss and this is especially affecting the network access measurements.

In all measurements, radio channel metrics were compared to gateway [27] and end device [17] sensitivity figures which are presented in Table 5. If the RSSI figures are above these sensitivity figures, a packets loss is not caused by radio characteristics.

Table 5. Gateway and End device sensitivity figures.

Device	**Signal bandwidth (kHz)**	**Spreading factor**	**Sensitivity (dBm)**
Gateway	125	7	-126
End device	125	7	-124

5.1 Measurement Accuracy

The evaluation platform accuracy is based on how well the diagnostic nodes are synchronized together. In order to measure such a performance, a timing measurement setup was build. As Figure 35 illustrates, all three end devices were set to transmit at same time stamp.

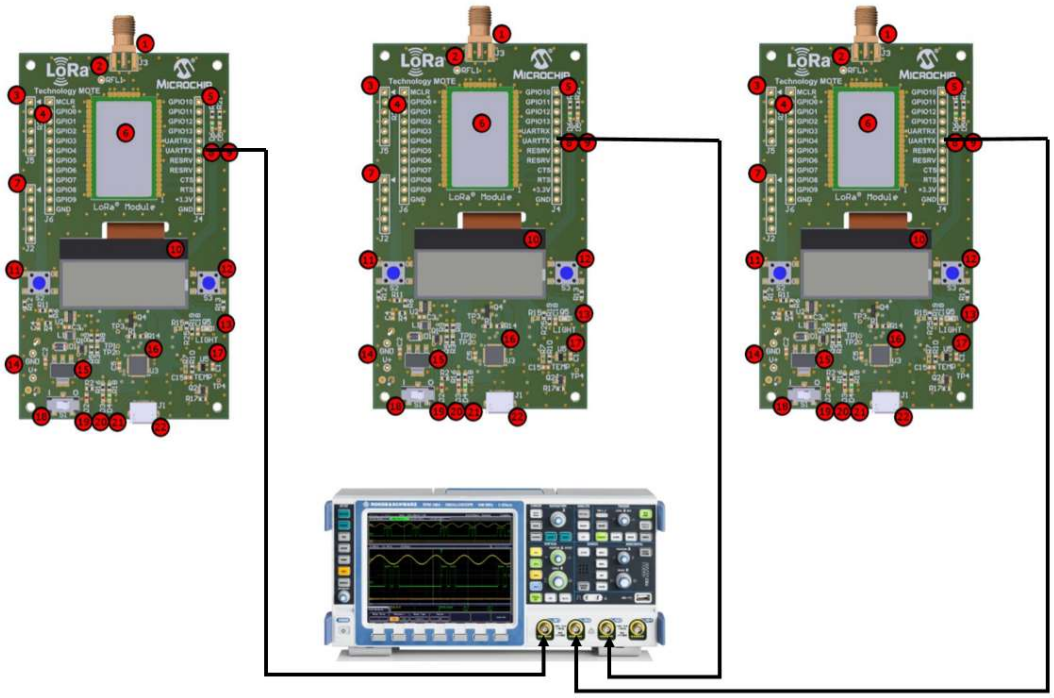

Figure 35. Accuracy measurement setup.

The measurements was carried out by measuring the end device Uart receive digital output pin. The transmitters are configured to transmit in same time stamp in other words all RDU's are using the same transmit json file as in Listing 6.

```
{
   "transmit_parameters": [
      {
         "status": "waitin_to_start",
         "send_forever": "false",
         "start_time": 1.52432947E9,
         "send_count": 1,
         "send_interval_milliseconds": 8000,
         "interval_decrement_milliseconds": 0,
         "data_content": "from_diagnostic_frame"
      }
   ]
}
```

Listing 6. Accuracy measurement JSON data structure.

So if the system is perfect the signals should rise up exactly in the same time, but real word measurement results are presented in Figure 36.

Figure 36. Accuracy measurement output.

The maximum delay difference between transmits signals were measured and the results are in Table 6.

Table 6. Evaluation platform accuracy measurement.

Measurement number	**Maximum delay difference (ms)**
1	4
2	4
3	9
4	4,5
5	2,5
6	10
7	4,7
8	6,6
9	2
10	5
Average	**5,2**
Average deviation	**1,8**

Better results would be achieved with a stratum 0 syncronization but current performance is sufficient for analysing the LoRaWAN network performance.

According to Figure 37 the radio channel characteristics were in suitable level, so the channel itself does not affect to measurement results.

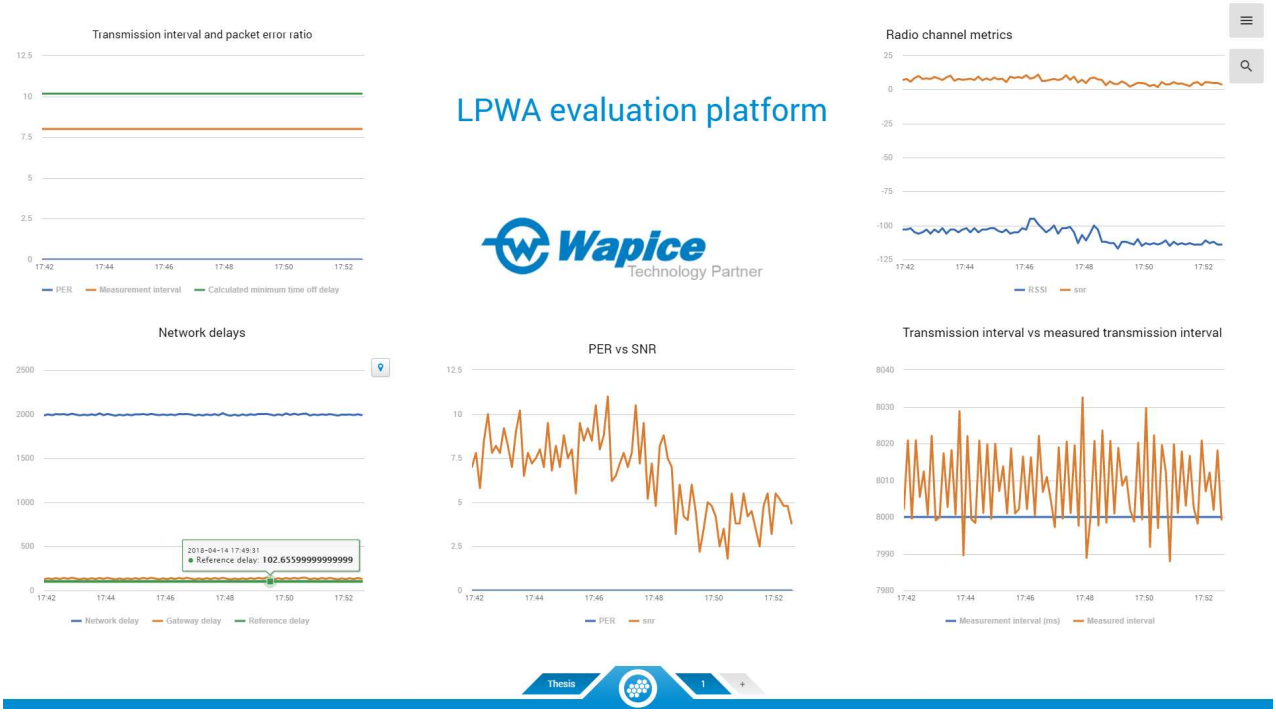

Figure 37. Radio channel characteristics during the measurements.

To ensure that the evaluation platform software performance is not affecting the measurement result the transceiver were set to transmit 7 hour test period and the test transmissions were logged to log file. This test was executed with 3178 repeats and the result is on table 7.

Table 7. Transmission count test.

Transmit count	**count of transmission's from log file**	**Transmission interval**	**Test period time**
3178	3178	8 s	7 h

As a conclusion the evaluation platform software functionality or radio channel performance is not affecting to measurement results.

5.2 Lowaran QoS Measurements

These measurements were executed according to listing number 7 Json content

```
self.data = {}
        self.data['transmit_parameters'] = []
        self.data['transmit_parameters1'] = []
        self.data['transmit_parameters'].append({
            'start_time': 1523375100.3362799,
            'send_interval_milliseconds': 8000,
            'send_count': 10000,
            'send_forever': 'true',
            'status': 'waitin_to_start',
            'interval_decrement_milliseconds': 100,
            'data_content': 'from_diagnostic_frame'
        })
```

Listning 7. Qos measurement JSON data structure.

With this setup the RDU was sending a test packet with decreasing transmission intervals. Each interval was decreased by 100 milliseconds so the transmission's were executed as presented in figure 38.

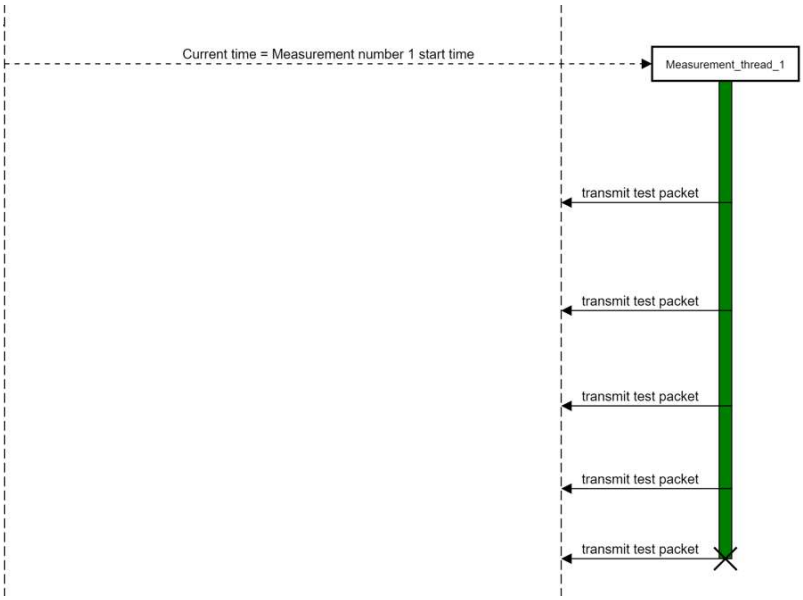

Figure 38. Qos decreasing transmission interval transmission test.

Results can be seen on Figure 39. According to PER readings, even the radio channel metrics were sufficient for error free transmission, the data was lost when the transmission interval reached a certain level. This is due to the duty cycle restriction. In order to follow the radio legislation duty cycle rules, the end device starts to prevent transmissions and that is the reason why packets were lost. This can be seen from the transmission interval versus measured transmission interval graph. It seemed that when the channel is blocked due to duty cycling the current test packet is waiting until channel is free and other packets, which RDU is trying to send, are lost.

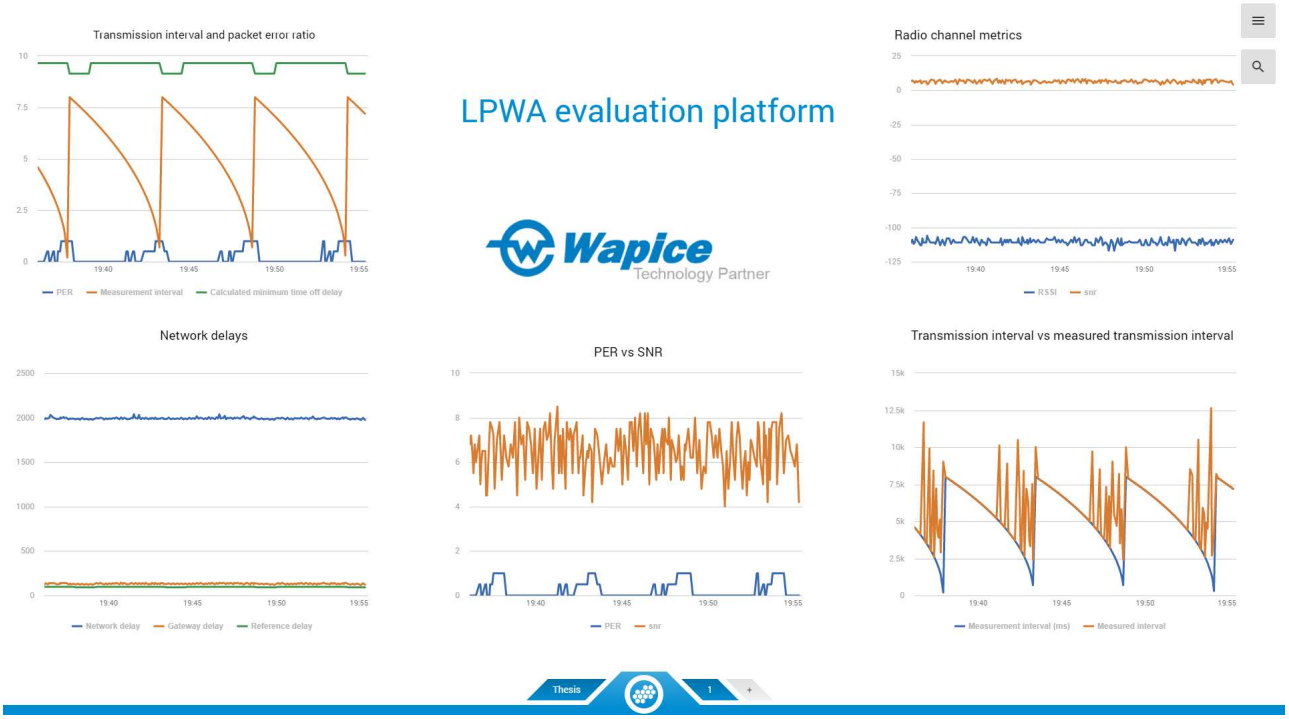

Figure 39. LoRaWAN short term measurements.

So, with this simple decrement measurement, the flaws of the implementation and LoRaWAN restrictions were detected and analysed.

5.2.1 Analysing Network Latencies

According figure 40, the LoRaWAN throughput average is about 2 seconds. The delay between the end device and the gateway was about 136 ms and during the measurement period this delay was stable. However, the 2000ms delay between the network

server and the end node was significant. This is not a sufficient performance for most of the industrial time critical applications.

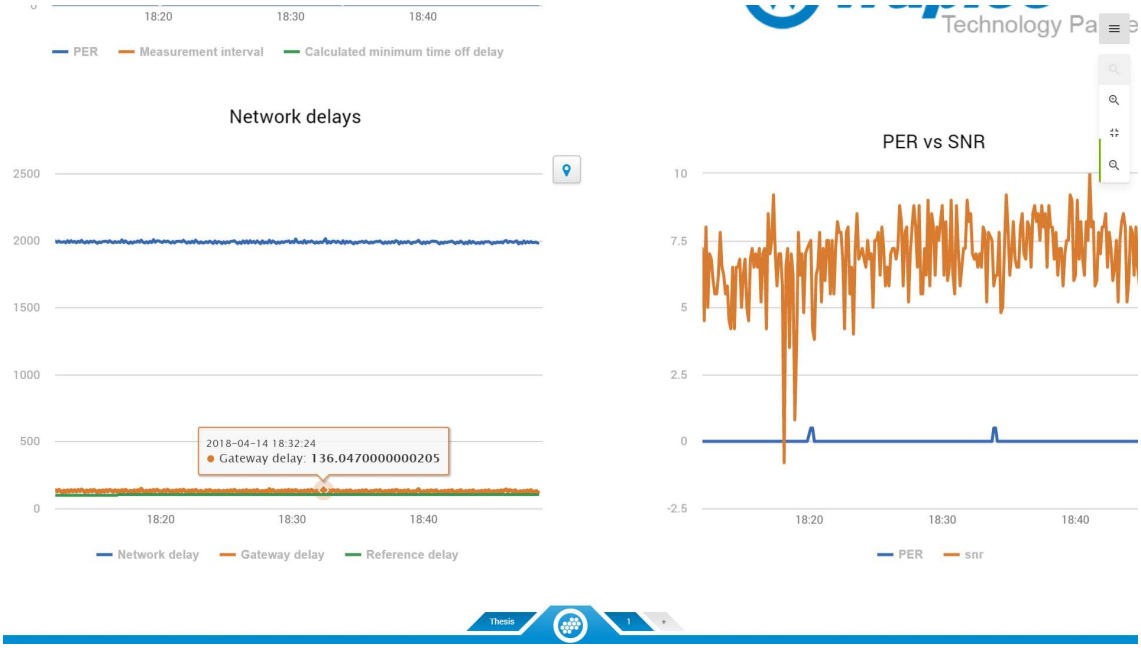

Figure 40. Network delays.

Detailed air interface performance is analysed in Figure 41 without an air interface offset. The transmission delay minimum and maximum values should be between the minimum and maximum reference values. Even the 5 ± 2 ms measurement error is removed from measurement result there is still over 15 to 20 ms difference between the theoretical and measured result.

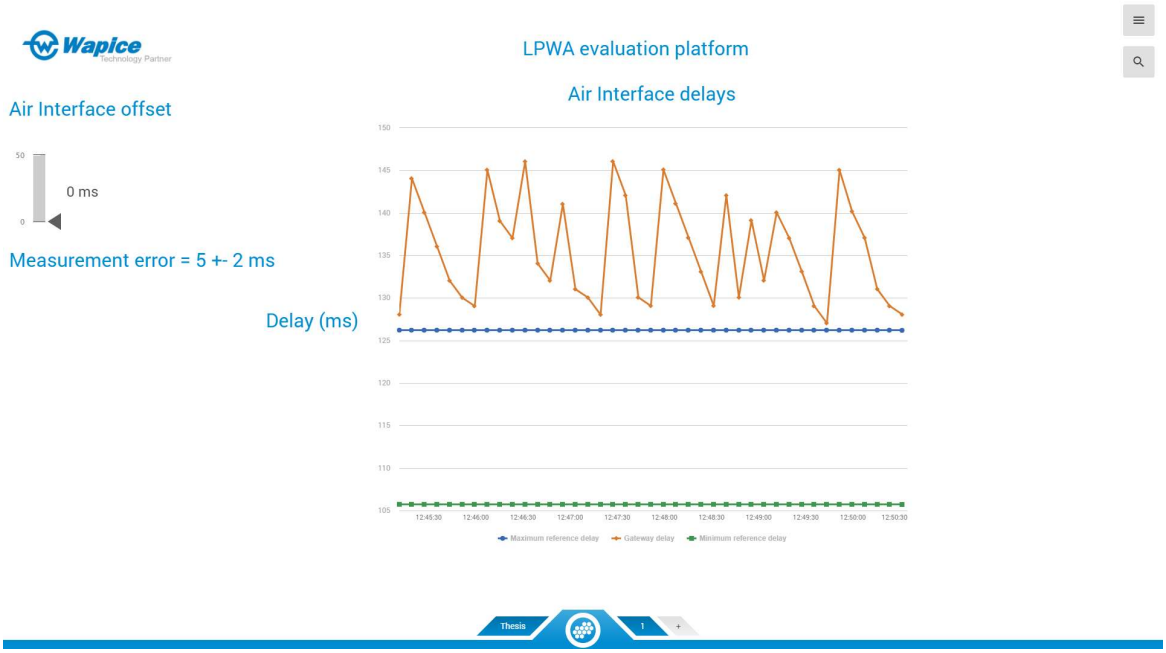

Figure 41. Air interface delay analysis without air interface offset

As Figure 42 illustrates, the 20 ms delay offset was added to measurement. Now the air interference delays were between minimum and maximum reference values. The air interface offset means that an offset value was added to reference delay values. Offset value is used to visualize the difference between the measured and calculated mean delay value. Additionally the analyzing of minimum and maximum delay is easier to apply since delay values has to fit between reference lines.

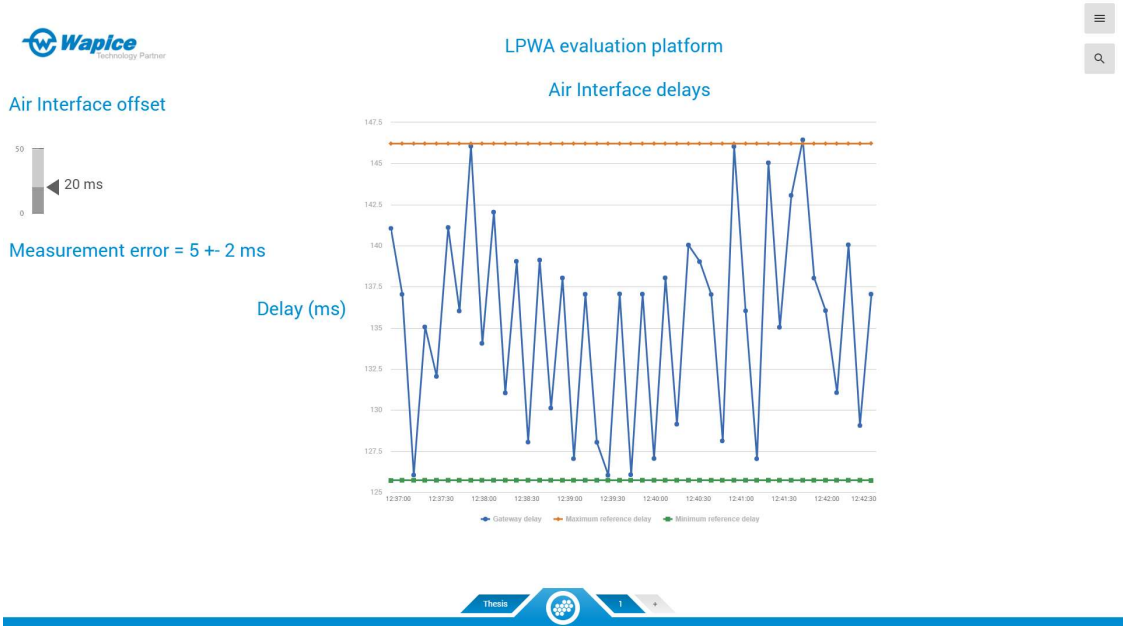

Figure 42. Air interface delay analysis with air interface offset.

So there was at least 15 ms excessive delay in air interface but with offset delay the measured delays are between the minimum and maximum values. This means that the delay variation due to network control is in acceptable level but the transmission delay does not correspond to calculated delays.

5.2.2 Analysing Multiple Channel Access

As illustrated in Figure 43, multiple channel access was analyzed. First, three end devices were set to transmit so that the transmission did not overlap. Transmission interval was 8 seconds in this case.

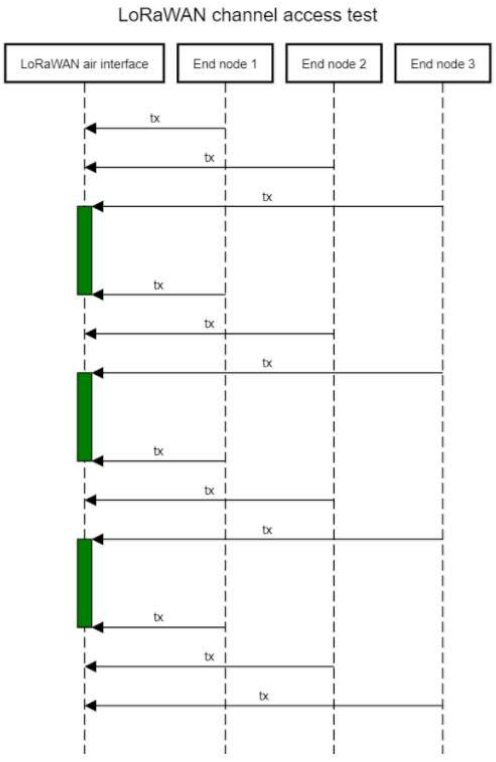

Figure 43. Multiple channel access test setup.

Figure 44 shows the result of this test. The operation of three end devices seems to be normal, transmission interval delays varies only same amount as earlier measurement.

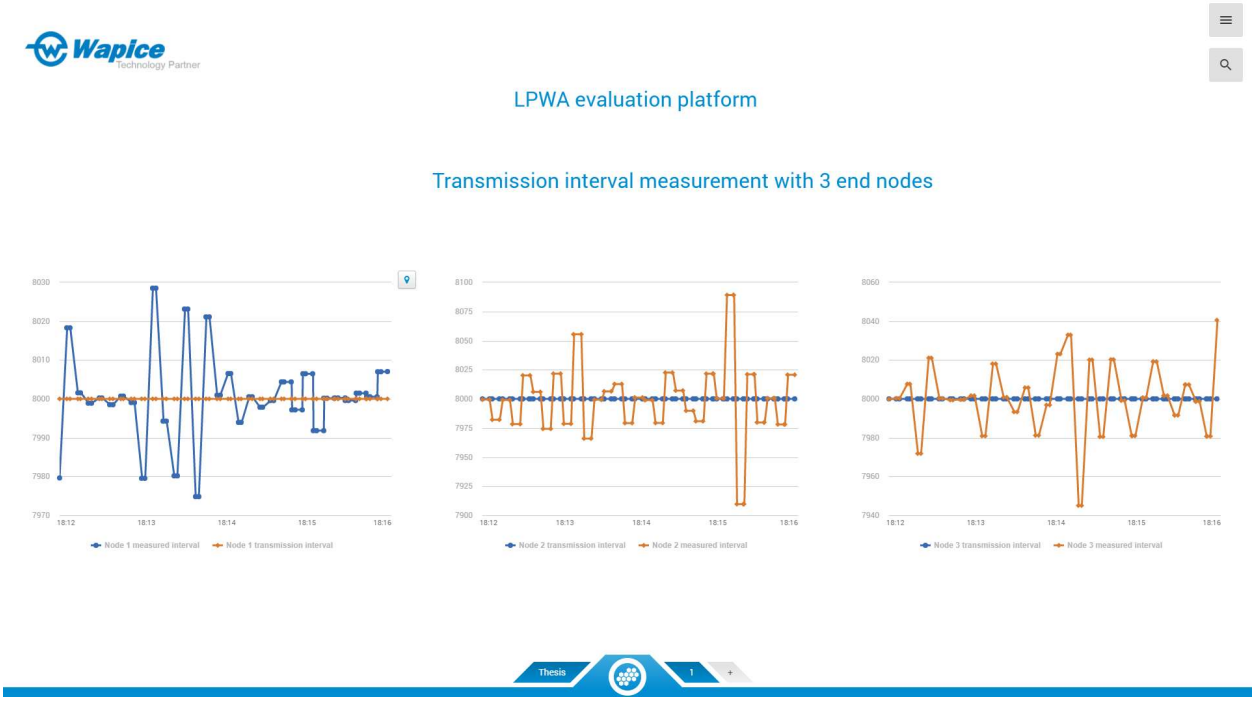

Figure 44. Multiple channel access test without colliding packets.

Figure 45 shows the next the transmission setup was changed so that the end devices transmissions collide in every transmission interval.

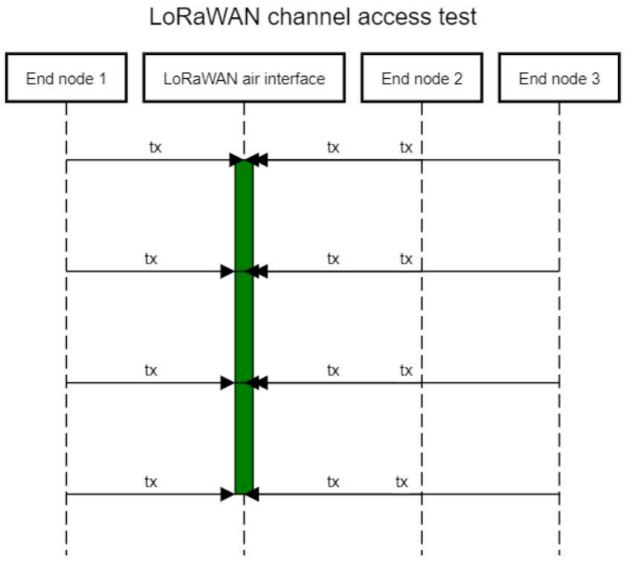

Figure 45. Collision test.

Test result is on Figure 46. Every now and then packets are lost which can be seen as a very long transmission interval. Actually, the measured transmission interval is a delay between the previous successful transmission and the current successful delay. So if

one packet is lost between those transmissions it is defined as doubled transmission delay. According to Figure 45, there can be two or three lost packets in a row. In this case, the delay is three or four times longer than in a normal transmission interval.

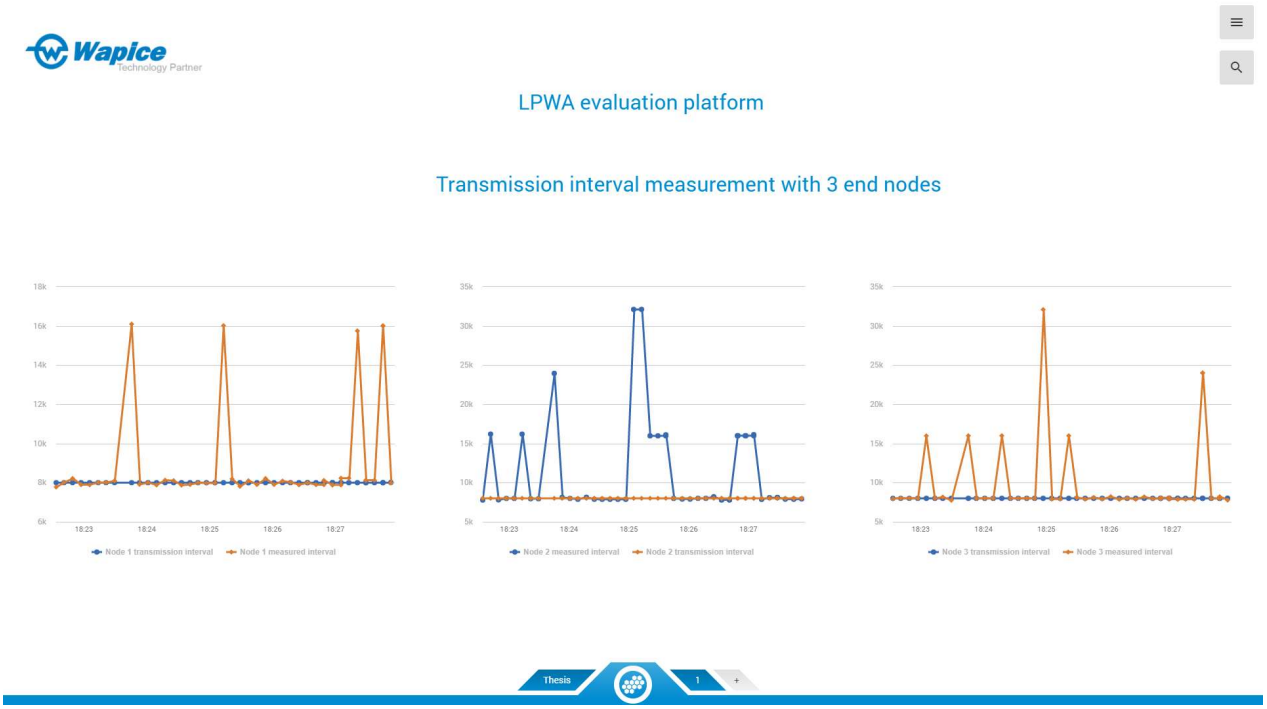

Figure 46. Multiple channel access test with colliding packets.

So according to measurement result the data retry must be used in order to maintain decent availability when network load increases. It might be that the use of ADR could also set the spreading factor to more interference tolerant setup in this colliding packets case.

5.3 Transmission Delay ECDF Curves

Previous measurements presented a short time frame of LoraWAN network. With ECDF curves the large amount of samples and a long time period can be analysed.

Now the actual measurement were executed according to Listning number 8 Json structure.

```
self.data = {}
        self.data['transmit_parameters'] = []
```

```
self.data['transmit_parameters1'] = []
self.data['transmit_parameters'].append({
    'start_time': 1523375100.3362799,
    'send_interval_milliseconds': 8000,
    'send_count': 1000,
    'send_forever': 'true',
    'status': 'waitin_to_start',
    'interval_decrement_milliseconds': 0,
    'data_content': 'from_diagnostic_frame'
})
```

Listning 8. ECDF measurement JSON data structure

This means 2 hour and 20 minutes testing period with a continuous transmission and every transmission interval was constant so the duty cycling should not interfere the measurement.

5.3.1 Gateway and Network Jitter Measurement's

First the network and gateway transmission delays were analysed. According to Figure 47, with 1000 sample measurement, 90% of the network delays samples were faster than 2002 ms. Maximum delay was 2075 ms and minimum was 1977 ms.

With same amount of samples, 90% of gateway delays were faster than 142 ms and maximum delay was 152 ms and minimum was 119 ms.

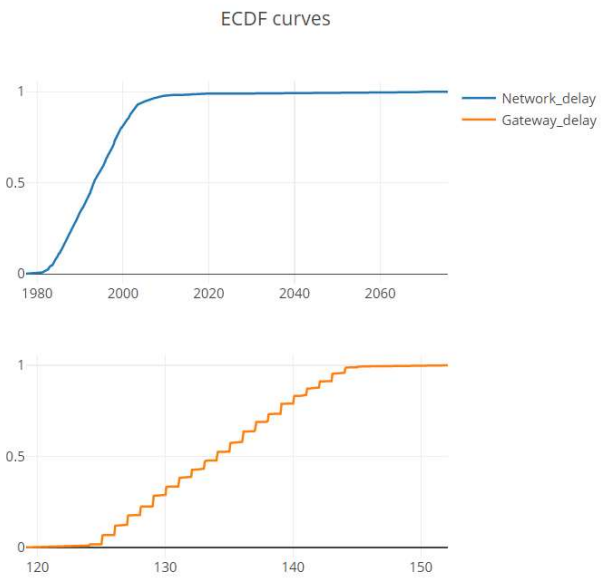

Figure 47. Gateway and network delay ECDF curves.

It seems that most of the jitter is caused by air interface. When this result is compared with chapter 5.2.1 measurement result, the jitter is almost the same as the difference of air interface minimum and maximum delay. So as a conclusion most of the jitter consist of network control message delay variation.

5.3.2 Availability Analysis

This measurement had the same setup as in previous chapter, so the RDU was sending with 8 second interval. In this measurement the transmission interval was under observation.

As the Figure 48 illustrates, 99.69 % of measured transmission interval delays were under the reference value.

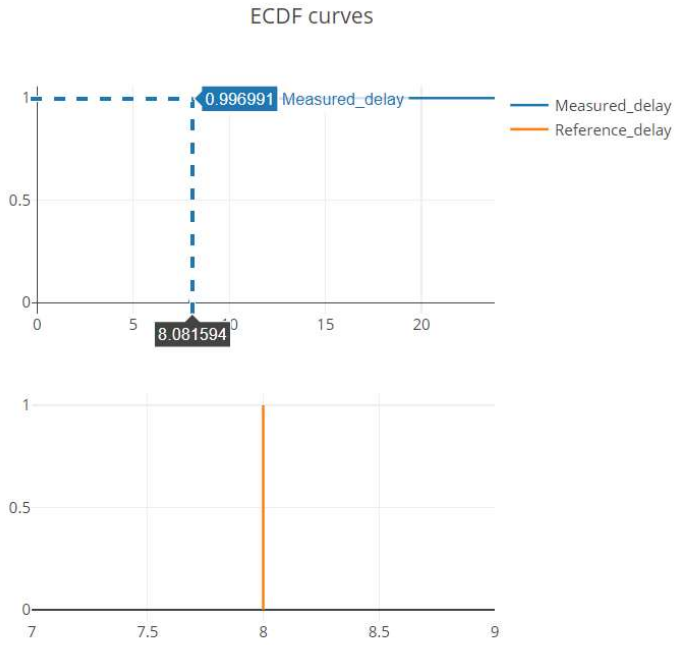

Figure 48. ECDF Availability analysis.

In Figure 49 99.89 % of the measurement delays were under 16 seconds. There was also one sample with 32 seconds.

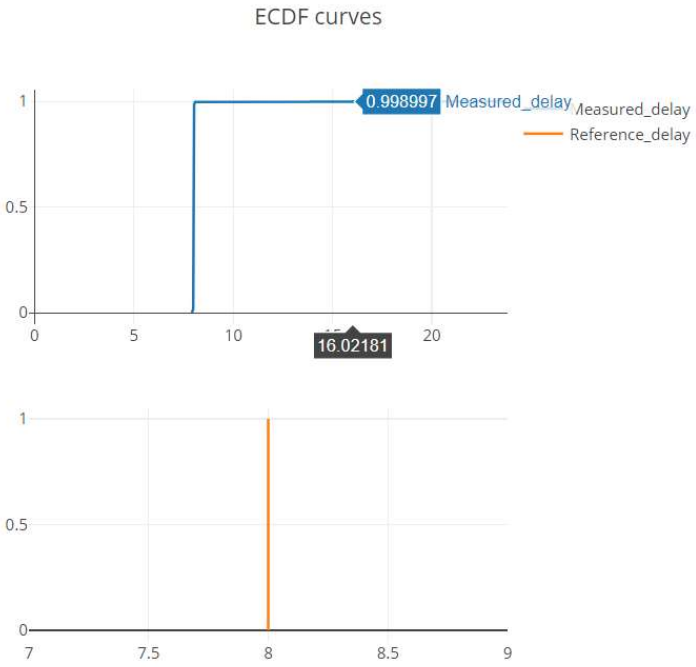

Figure 49. Availability ECDF curves.

This is a clear indication of lost packets, which can be seen also on Figure 50 PER figures.

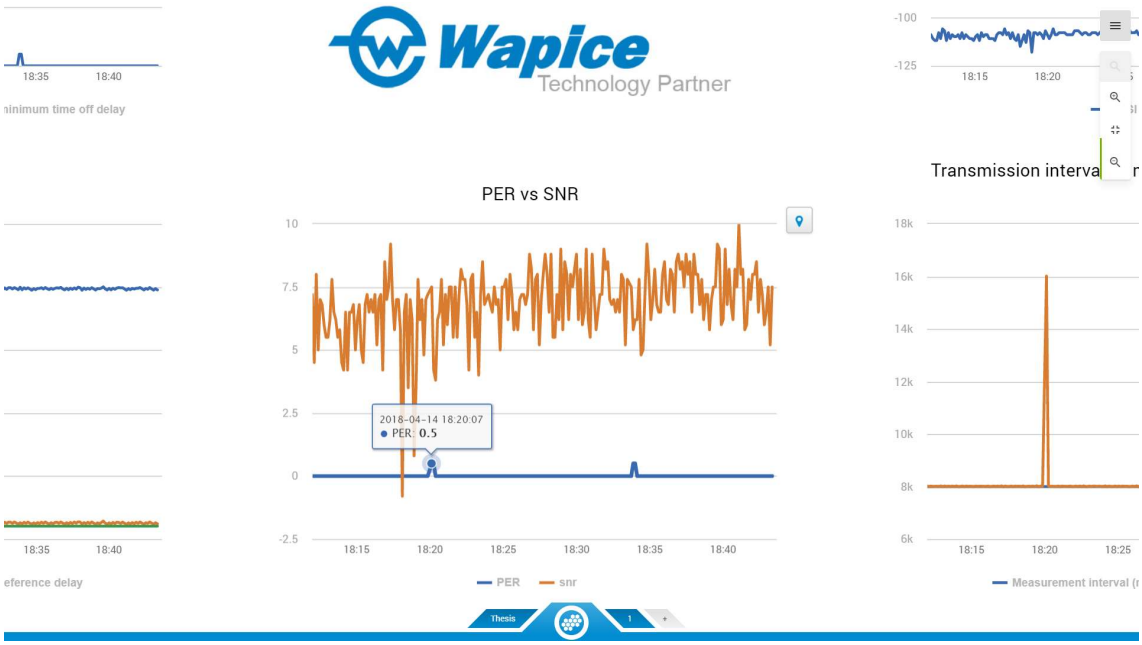

Figure 50. Availability measurement PER vs SNR graph.

As a conclusion, with this network setup the LoraWAN network performance does not correspond four nines availability requirement. However, the measurement was carried out without retry and ADR mechanism. So even the RSSI is well above the sensitivity level this system requires to use the data retry.

6 Conclusion

According to chapter 5 analysis, with the developed LPWA evaluation platform the flaws and weaknesses of the LPWA technologies and networks components can be analyzed in a comprehensive way. It is also possible to isolate whether the problems are related to radio channel, LPWA technology restriction or a single network component. The measurement system was designed such as a true IoT system and it is capable to analyze its communication components.

It was surprising how easy it is to build a measurement system with open source hardware and software. Rpi with Python libraries provides powerful tools for generating test data and analyzing test results. The IoT ticket interface was also easy to implement. Only problem was the interface between the network server and the IoT ticket. Due to not compatible interface a network adapter was implemented. Also the IoT ticket does not support ECDF analysis so the plotly interface was implemented to the evaluation platform in order to analyze the statistics of the transmission.

The platform timing accuracy is sufficient for analyzing the LoRaWAN networks. If more precise results are required, RDUs should be connected directly to stratum 0 clock source. In addition, the platform software functionality did not affect the test results. The PER measurement was relying on CRC and a transmission count so if the test packets are arriving in different order this method performance is unsuitable. When a packet is lost there is no easy way to isolate what network element has lost the packet. Of course the radio channel characteristics can provide a hint is the problem in the air interface but for accurate information the PER measurement should be implemented to the gateway also.

Delay measurements analysis provided information about the jitter, availability and delay of individual network components. Furthermore, the LoRaWAN protocol control messages and evaluation platform error were considered when jitter values were evaluated. As a conclusion, the LoRaWAN network server delay was dominating the overall delay, so some optimization should be performed in order to archive better overall delay performance. According to availability analysis, without ADR and data retry LoRaWAN network does not fulfill four nines criteria. Multiple channel access requires also ADR and data retry, otherwise data is lost due to collisions. Finally, the jitter performance was following theoretical reference values.

As a conclusion, this evaluation platform is a powerful tool for analyzing LPWA technologies but it has a weakness in packet error ratio measurement that should be improved in the next generation development. Collecting diagnostic data from all network actuators would provide accurate information about the network status as shown in Figure 51.

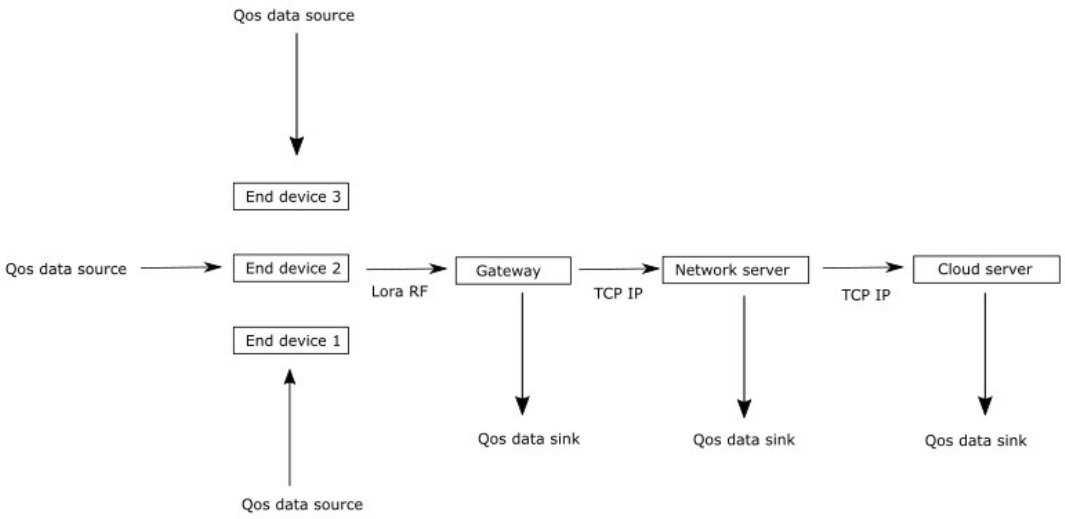

Figure 51. Next generation evaluation platform.

This would require access to the gateway and network log files. This was not possible to implement with current evaluation platform network components. So network designer should consider the logging capabilities as a critical network component selection criteria.

Next logical step would be to implement the NB-IoT measurement to the LPWA evaluation platform. Then the competing technologies, LoraWAN and NB-IoT could be compared side by side with real time measurements and the results could be analyzed with IoT ticket visualization capabilities.

Finally, this evaluation platform could be used to test different communication systems than LPWA networks. For example studying Bluetooth communication system with an evaluation platform could reveal issues in advance before actual implementation. One might even consider using this platform as a basis of an IoT application to evaluate network performance and detect possible design problems in an early stage of a development cycle.

References

1 Lora alliance, LPWA Technologies: Unlock new IoT Market Potential, white paper.https://docs.wistatic.com/ugd/eccc1a_a45b82212b3e4a3098cc714200e8e5c6.pdf. Accessed September 3 2017.

2 Sinha, R.S., Wei, Y. & Hwang, S.-H. A survey on LPWA technology: LoRa and NB-IoT. ICT Express. 2017

3 Lora alliance 2017. LORAWAN, What is it?, white paper. https://docs.wix-static.com/ugd/eccc1a_ed71ea1cd969417493c74e4a13c55685.pdf. Accessed September 9 2017.

4 Margelis, G. Low Throughput Networks for the IoT: Lessons learned from industrial implementations. Internet of Things (WF-IoT). 2015

5 Zayas, Almudena Diaz. The 3GPP NB-IoT System Architecture for the Internet of Things. Communications Workshops (ICC Workshops), 2017 IEEE International Conference. 2017.

6 R. B. Sørensen, D. M. Kim, J. J. Nielsen and P. Popovski, "Analysis of Latency and MAC-layer Performance for Class A LoRaWAN," in IEEE Wireless Communications Letters. 2017.

7 Cisco Press [Internet]. Cisco Press: Source for Cisco Technology, CCNA, CCNP, CCIE Self-Study. http://www.ciscopress.com/store/end-to-end-qos-network-design-quality-of-service-for-9781587143694. Accessed September 17 2017.

8 Mina Nabi, Maria Toeroe, Ferhat Khendek, Availability in the cloud: State of the art, In Journal of Network and Computer Applications, Volume 60. 2016.

9 Maichen, W, Digital Timing Measurements: From Scopes and Probes to Timing and Jitter. 2006.

10 Yufeng Deng, S. A review of network latency optimization techniques. Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on. 2013.

11 Lim Su Jin, Simon. "Link Quality Prediction for Multimedia Streaming Based On Available Bandwidth and Latency." Local Computer Networks Workshops (LCN Workshops). 2013.

12 Lieverdink P. Infrastructure Services: NTP, DNS, DHCP, and SSH. Pro Linux System Administration. Apress. 2009.

13 M. Rizzi, P. Ferrari, A. Flammini and E. Sisinni, "Evaluation of the IoT LoRaWAN Solution for Distributed Measurement Applications," in IEEE Transactions on Instrumentation and Measurement. 2017.

14 F. Wang and X. Wang, "Fast and Robust Modulation Classification via Kolmogorov-Smirnov Test," in IEEE Transactions on Communications, 2010.

15 Arslan, Hüseyin. Cognitive Radio, Software Defined Radio, and Adaptive Wireless Systems. 2007.

16 N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. Hersent, LoRaWAN Specifications. LoRa Alliance. 2016

17 Semtech, SX1272/73 datasheet http://www.semtech.com/images/datasheet/sx1272.pdf . Accessed september 27 2017.

18 Augustin, A. Study of LoRa: Long Range & Low Power Networks for the Internet of Things. Sensors. 2016

19 LoRa Alliance. Technical committee, LoRaWAN 1.0.2 Regional Parameters. 2016.

20 F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui and T. Watteyne, "Understanding the Limits of LoRaWAN," in IEEE Communications Magazine. 2017.

21 ETSI. Electromagnetic compatibility and Radio spectrum Matters (ERM); Short Range Devices (SRD); Radio equipment to be used in the 25 MHz to 1 000 MHz frequency range with power levels ranging up to 500 mW; Part 1: Technical characteristics and test methods. 2012.

22 Semtech, Lora design quide https://www.semtech.com/uploads/documents/LoraDesignGuide_STD.pdf. Accessed April 14, 2018.

23 Wapice, Iot ticket platform. https://www.iot-ticket.com/platform. Accessed September 21 2017.

24 Aihkisalo, Tommi. "Latencies of Service Invocation and Processing of the REST and SOAP Web Service Interfaces." Services (SERVICES), 2012 IEEE Eighth World Congress On. 2012.

25 IMST Wimod lite gateway quick start guide. https://wireless-solutions.de/images/stories/downloads/Evaluation%20Tools/LoRa_Lite_Gateway/WiMOD_LiteGateway_QuickStartGuide_V1_1.pdf. Accessed April 14, 2018.

26 P. S. Cheong, J. Bergs, C. Hawinkel and J. Famaey, Comparison of LoRaWAN classes and their power consumption. 2017 IEEE Symposium on Communications and Vehicular Technology. 2017.

27 IMST. Wimod datasheet. https://cdn.sos.sk/productdata/29/eb/a68245ed/im880b-l-lorawan.pdf, Accessed April 14, 2018.

28 Loriot, product web page, https://www.loriot.io/pricing.html. Accessed April 14, 2018.

29 Lora-net. Open source Lora gateway project, https://github.com/Lora-net/lora_gateway, Accessed April 14, 2018.

30 Microchip. LoRa® Mote User's Guide. http://ww1.microchip.com/downloads/en/DeviceDoc/LoRa%20Mote%20Users%20Guide.pdf. Accessed April 14, 2018.

31 Microchip. Low-Power Long Range LoRa® Technology Transceiver Module. http://ww1.microchip.com/downloads/en/DeviceDoc/50002346C.pdf. Accessed April 14, 2018.

32 Microchip. RN2483 Lora technology module command reference user's guide. http://ww1.microchip.com/downloads/en/DeviceDoc/40001784B.pdf. Accessed April 14,2017.

33 Numpy. Web pages.http://www.numpy.org/. Accessed 9 May 2018.

34 Plotly. Plotly for Python. https://plot.ly/d3-js-for-python-and-pandas-charts/. Accessed 9 May 2018.

IoT ticket interface

```
rssi = data["gws"][0]
float_rssi = float(rssi["rssi"])
nv = datanodesvalue()
nv.set_name("RSSI")
nv.set_path(iot_id)
nv.set_dataType("double")
nv.set_value(float_rssi)
nv.set_timestamp(timeStamp)

gateway_delay = result['gatewaydelay']
float_gvd = float(gateway_delay)
nv1 = datanodesvalue()
nv1.set_name("Gateway delay")
nv1.set_path(iot_id)
nv1.set_dataType("double")
nv1.set_value(float_gvd)

nv1.set_timestamp(timeStamp)

network_delay = result['networkdelay']
float_nvd = float(network_delay)
nv2 = datanodesvalue()
nv2.set_name("Network delay")
nv2.set_path(iot_id)
nv2.set_dataType("double")
nv2.set_value(float_nvd)
nv2.set_timestamp(timeStamp)

per = result['per']
float_per = float(per)
nv3 = datanodesvalue()
nv3.set_name("PER")
nv3.set_path(iot_id)
nv3.set_dataType("double")
nv3.set_value(float_per)
nv3.set_timestamp(timeStamp)

reference_delay = result['calculated_delay_min']
float_reference_delay = float(reference_delay)
nv4 = datanodesvalue()
nv4.set_name("Minimum reference delay")
nv4.set_path(iot_id)
nv4.set_dataType("double")
nv4.set_value(float_reference_delay)
nv4.set_timestamp(timeStamp)
c = Client(self.baseurl, self.username, self.password)

print(c.writedata(self.dev_iot_id, nv, nv1, nv2 ,nv3 nv4))
```

Result calculation method

```
def calc_result(self,input):

    # calculating delays from timestamps
    Ts = datetime.datetime.now()
    end_node_delay = input[0]
    network_server_delay = Ts.microsecond + Ts.minute + Ts.second
    if self.previous_time_stamp != 0:
            measured_time_interval=network_server_delay - self.prevous_time_stamp
    self.previous_time_stamp = network_server_delay
    gateway_server_delay = self.calc_ms(input[1])
    self.data['gatewaydelay'] = gateway_server_delay - end_node_delay
    self.data['networkdelay'] = network_server_delay - end_node_delay
    self.data['interval'] = input[3]
    self.data['measinterval'] = measured_time_interval
    self.data['interval_ms'] = input[3] * 1000
    self.data['calculated_delay'] = self.calc_ref_delay(input)

    # storing the long term data for cdf calculation
    if not self.network_delay_logfile.addTxData(self.data['networkdelay']):
            self.network_delay_logfile.strore_data()
    if not self.gateway_delay_logfile.addTxData(self.data['gatewaydelay']):
            self.gateway_delay_logfile.strore_data()
    if not self.availability_delay_logfile.addTxData(input[3]*1000):
            self.availability_delay_logfile.strore_data()
    if not self.availability_delay_reference_logfile.addTxData(meas_time_inteval):
            self.availability_delay_reference_logfile.strore_data()

    #PER calculator
    self.data['per'] = self.per_calculator(input[2])

    return self.data
```

Reference delay calculation methods

```
def calc_ref_delay(self,input):

    sf = input[4]
    bw = input[5]
    de = 0
    CR = input[6]
    PL = input[7] + 15
    H = 1
    n_preample = 8
    tsym = math.pow(2, sf) / bw
    Tpreample = (n_preample + 4.25) * tsym
    symbolcount = 8 + max(math.ceil((8 * PL + 4 * sf + 28 + 16 - 20 * H) / (4 * sf - 2 * de)) * (CR + 4), 0)
    T_payload = symbolcount * tsym
    total = 1000*(T_payload + Tpreample)
    return total

def minumum_off_period_time(self,time_on_air):
    duty_cycle = 0.01
    minimum_off_period_time = time_on_air*((1/duty_cycle) - 1)
    return minimum_off_period_ti
```

ECDF graph plot method

```
from file_handler import file_hander
from Graph import graph
import sys

def main():

    cdf_graph = graph()
    print("*** CDF tool *** usage cdf filename1 cdfname1 filename2 cdfname2 \n")
    for arg in sys.argv:
        print("args: ",arg)
    file_name = sys.argv[1]
    file_name1 = sys.argv[3]
    handler = file_hander(file_name,0)
    handler1 = file_hander(file_name1,0)
    data=handler.read_file()
    data1 = handler1.read_file()
    cdf_graph.cdf(data, data1, sys.argv[2], sys.argv[4])


if __name__ == "__main__":
    main()
```

Graph class with numpy and plotly python libraries

```
import plotly
from plotly import tools
from plotly.graph_objs import *
import plotly.plotly as py
import numpy as np

class graph:
    def __init__(self):
        plotly.tools.set_credentials_file(username='xxxx', api_key='qwertyyuuiip')

    def cdf(self, datain, datain2, name1, name2):

        # compute the  first CDF
        cdfx = np.sort(datain)
        cdfy = np.linspace(1 / len(datain), 1.0, len(datain))
        # plot the CDF
        trace1 = Scatter(
            x=cdfx,
            y=cdfy,
            name=name1,
        )
        # compute the  second CDF
        cdfx2 = np.sort(datain2)
        cdfy2 = np.linspace(1 / len(datain2), 1.0, len(datain2))
        # plot the CDF
        trace2 = Scatter(
            x=cdfx2,
            y=cdfy2,
            name=name2,
        )

        fig = tools.make_subplots(rows=2, cols=1)
        fig.append_trace(trace2, 1, 1)
        fig.append_trace(trace1, 2, 1)
        fig['layout'].update(height=600, width=600, title='CDF curves')
        py.plot(fig, filename='stacked-subplot')
```