VRIJE UNIVERSITEIT AMSTERDAM

Assignment

Fundamentals of Adaptive Software 2022-2023

Assignment: Implementation and Demo
Group: DingNet-David-3

Member names:Alina Boshchenko, Dmitriy Knyajev, Nichita Goncear, Dmitrijs Voronovs
Emails: (a.boshchenko@student.vu.nl, d.knyajev@student.vu.nl, n.goncear@student.vu.nl, d.voronovs@student.vu.nl)
VUnetIDs: (2732782, 2770777, 2760519, 2779206)
Master program and track: Computer Science - Green IT

December 27, 2022

Contents

1. Proposal

1.1. The uncertainty that the extension addresses

NAME	Ambiguity
CLASSIFICATION	Run-Time
CONTEXT	Ambiguity occurs when devices lack precision and accuracy in a measurement and have a decreased reliability in terms of communicating.
IMPACT	This source of uncertainty can affect the state of the application when certain devices become less reliable, resulting in the package losses that are not reflected in the simulations and in graphs.
DEGREE OF SEVERITY	High, since failure to implement an effective adaptive strategy can reduce the reliability of the system

Table 1: Brief description of uncertainty

1.2. The need for/importance of addressing this uncertainty

Studies[1] show that with age, the accuracy of sensors deteriorates, which leads to incorrect actions when processing reports, failures, false positives, or vice versa in certain types of sensors. Sensors are exposed to various influences that accelerate aging during their life cycle, which leads to bigger message loss, weaker signal, and lower precision in measurements. Without an adaptive strategy that handles these factors, over time, the data received from the sensors will no longer be accurate, which negates the effectiveness and expediency of using the system.

1.3. The extension of the adaptation logic

Digital circuit reliability and performance of the device are negatively impacted by IoT device aging. The adaptation logic addresses this issue by gradually increasing the power settings to keep the device available for communication with a gateway. According to other adaptation strategies, the spreading factor and the sampling rate are maintained constant. Normally, they are set up through a process of trial and error, which is outside the scope of the task.

1.4. The expected effects of using the extension

The proposed adaptation logic will have a positive effect on the percentage of data packages lost. Applying the SAS extension will decrease the quantity of lost packages, resulting in uniform data with minimal knowledge gaps.

By manipulating power settings and the frequency of data packages sent, the system will increase its reliability, although it comes at a higher cost of energy consumption.

2. Analysis and design

2.1. Overall system description

DingNet is a network system that develops an IoT research environment for smart city applications. The system works predominantly in the city of Leuven, Belgium, and supports deployment and monitoring of large-scale applications, consisting of a significant number of remote devices called motes.

The system is built upon the LoRaWAN network, which is a communication protocol designated to allow long-range IoT device communication. The network consists of multiple static and mobile motes throughout the city. The motes gather knowledge about their environment and transmit it to the gateways within communication range. There are 14 gateways [2] deployed on high buildings to provide better coverage and signal strength with the motes.

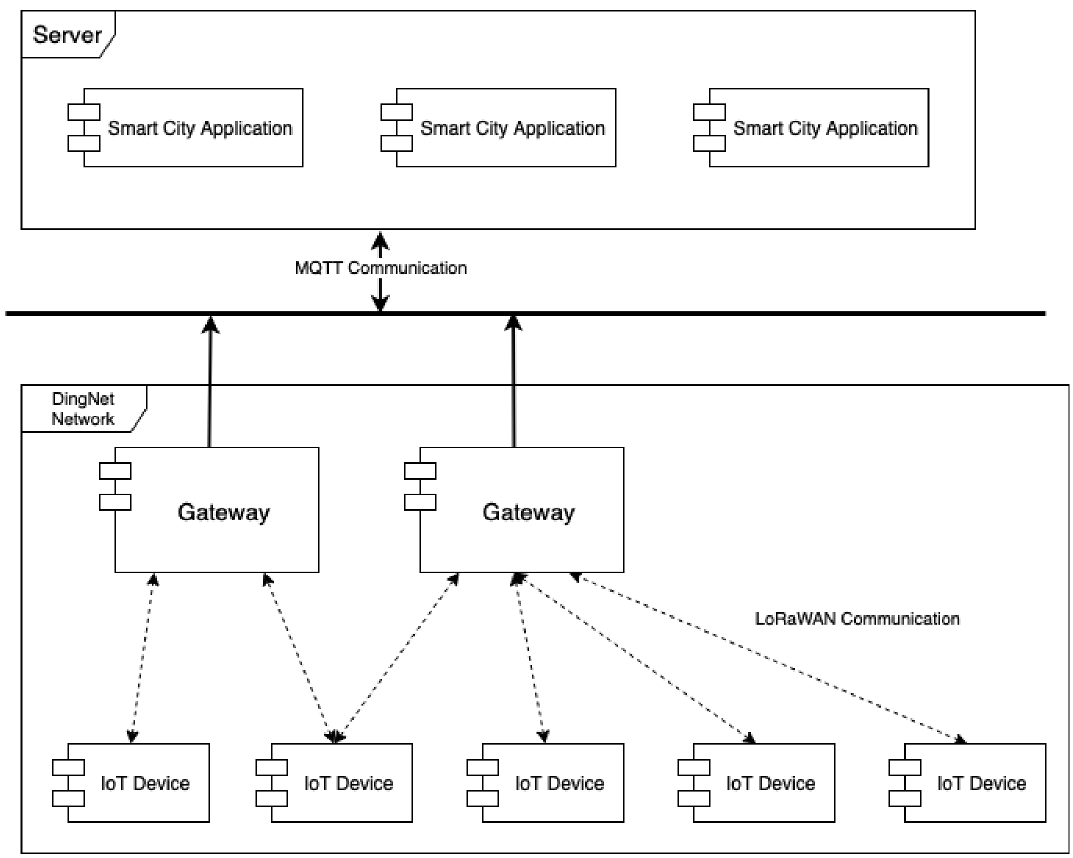


Figure 1. DingNet architecture

Some potential applications of the DingNet network are: garbage bins that notify the administrator when they need to be emptied, air quality tracking devices, parking availability sensors, etc.

To help deal with the challenges of developing IoT applications for smart cities, the DingNet comes with a GUI simulator. The simulator allows for systems' preliminary testing before their deployment. The simulator comes with a map of Leuven and multiple customization opportunities. The map displays the gateways, motes, and their paths, in case those are mobile. The user can control the number, distribution, and trajectory of mobile motes.

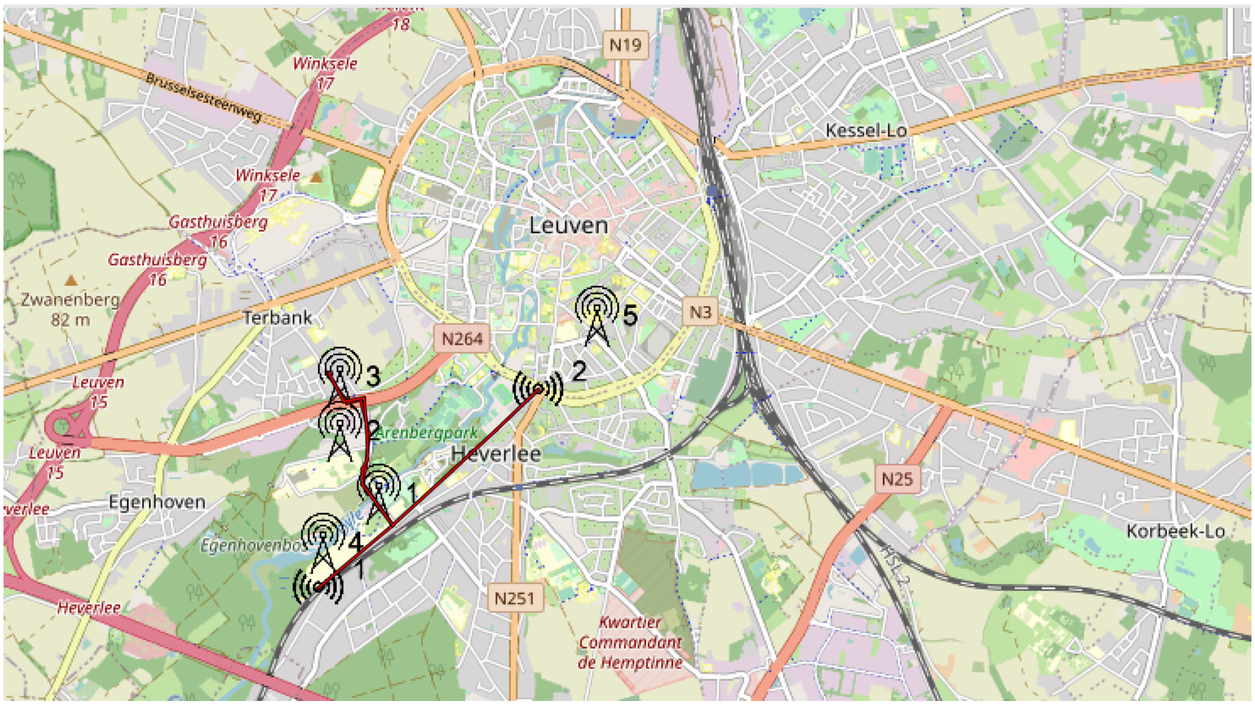


Figure 2. Map view of the DingNet simulator

After performing a single or multiple runs, mobile motes reach their destination point and gather all the environmental data. Some of the motes' tracked properties are: received power, distance to the gateway, power settings, and used energy. The data is displayed in graphs at the bottom of the screen.

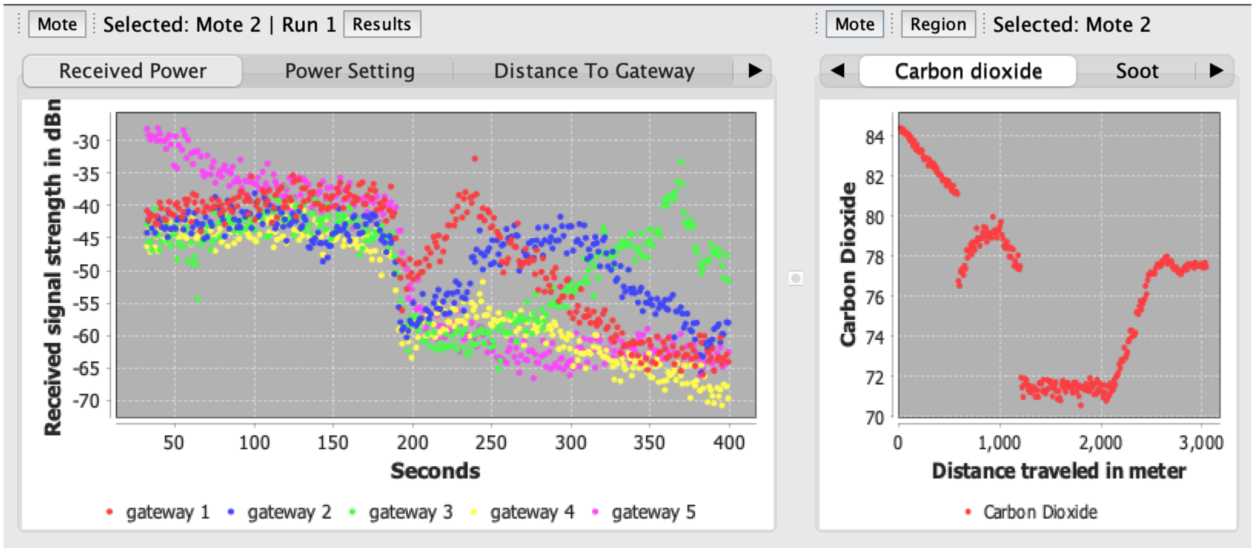


Figure 3. Motes' tracked technical properties

The system is subject to multiple uncertainties, some of which are addressed in the exemplar paper [2], which demonstrates how these can be handled with the help of the DingNet simulator. The discussed uncertainties are distance and signal strength at the gateway. The paper proposes distance-based adaptation to adapt mote's power setting proportionally to the distance to the nearest gateway. And the signal-based adaptation is meant to incrementally increase the mote's power settings if the signal strength goes below a minimum threshold. These are just a few of the opportunities provided by the DingNet simulator; additionally, the simulator's source code is open source, allowing users to customize it to their specific needs.

2.2. Analysis of uncertainties

NAME	Ambiguity
CLASSIFICATION	Run-Time
CONTEXT	Ambiguity occurs when devices lack precision and accuracy in a measurement and have a decreased reliability in terms of communicating.
IMPACT	This source of uncertainty can affect the state of the application when certain devices become less reliable, resulting in the package losses that are not reflected in the simulations and in graphs.
DEGREE OF SEVERITY	High, since failure to implement an effective adaptive strategy can reduce the reliability of the system
SAMPLE ILLUSTRATION	As a sample illustration, we can present the necessity to consider device aging in the conducted simulations. The reliability of the device decreases with age. By taking this factor into account and applying actions such as setting a probability of packet loss and a probability of mote repair, it is possible to operate the data with higher precision.
EVALUATION	SAS solution handling the defined uncertainty, in comparison to the non-SAS, allows better accuracy of the experiment and stat reflection in plots. As a metric, we can take the number of packets lost and energy consumption. With the adaptation implemented, the average number of package losses on the device should be lower, and the level of energy consumption should be tuned. As a drawback of the SAS approach, we can consider potentially higher energy consumption by the device in comparison to the non-SAS.
ALSO KNOWN AS	Imprecision

Table 1: Addressed uncertainty

In the current paper we would like to address the Ambiguity uncertainty. Our goal is to raise the reliability of the devices and reduce the ambiguity and imprecision. Not handling this uncertainty in our case will result in the package loss and less trustworthy simulation. To address the uncertainty we aim to handle the factor of motes aging. It is essential to adapt the system so that motes can function in a reliable way not only in the new condition, but also after the degrading effects of aging have occurred.

2.3. Requirements

In this section, a concise definition of system requirements for adaptive system is provided. The section is divided into two parts, traditional requirements, which are divided into functional (Table 2) and non-functional requirements (Tables 3-4), and RELAX requirements [3] (Tables 5-6) for adaptive systems. Since obtaining accurate data plays the main role for the correct operation of the system, the main attention to the formulation of requirements is given to it. System requirements also take sustainability into account. As it is incredibly important to take this area into account

Functional requirements

Identifier	NR1-DeviceAgeResponse
Description	The system must respond to the deterioration of the device, taking into account its age and adaptively adjust the values of the supplied energy to compensate for degraded performance.
Motivation	The age of the device can play a big role in the reliability of reading and transmitting data. If we ignore the fact of device aging, after some time of operation, the data will not be reliable.

Table 2: Explanation of Functional Requirement 1: DeviceAgeResponse

Non-Functional requirements

Identifier	NR2-Sustainability
Description	The system should not consume more than 500 mJ during the normal duration of the experiment (280 seconds), as larger values will result in overconsumption.
Motivation	Since IoT solutions are based on the principle of energy efficiency, and high power consumption can become critical in some situations (usage of battery power) - the system must be energy efficient during normal operation

Table 3: Explanation of Non-Functional Requirement 1:Energy Efficiency/Sustainability

Identifier	NR3-Reliability
Description	The system should ensure that the package loss rate is close to zero (less than 2%) to keep the system running optimally during the operation time. The system should increase power supply, based on the age of the

	device or other factors when it's obvious that these factors are affecting mote operation.
Motivation	Data plays a big role in the system, as most decisions are made based on If motes work and capture information, but cannot deliver it to the processing device, this means that the system is wasting resources and working to no avail. The system should provide reliable information for all stakeholders.

Table 4: Explanation of Non-Functional Requirement 2

RELAX Requirements

ID	RR-1
Statement	The system SHALL ensure that package loss rate is AS CLOSE AS POSSIBLE TO zero during the operation time. The system SHALL increase the power supply, based on the age of the device AS EARLY as it starts to affect the characteristics of the device.
Environment	Package loss rate.
Monitoring	Device aging coefficient, Device energy monitors.
Relations	Device aging coefficient and package loss rate can be used to increase the reliability of the system.
Grammar Expression	SHALL (AS CLOSE AS POSSIBLE, AS EARLY)

Table 5: Explanation of RELAXed Requirement 1

ID	RR-2
Statement	The system SHALL consume AS FEW units of energy AS POSSIBLE during normal operation.
Environment	Total energy consumption.
Monitoring	Device energy monitors.
Relations	Device energy monitors can sense device energy consumption and sense activity within the device, and use these to control update intervals.

Grammar Expression	SHALL (AS FEW AS POSSIBLE)

Table 6: Explanation of RELAXed Requirement 2

2.4. Analysis of potential solutions

Aging is one of the major reliability threats, especially in safety-critical applications for instance in the automotive domain [4]. Device aging results in performance degradation and eventual failure of digital circuits over time [5].

The major factors we aim to address in the proposed solution are initial age, total energy consumption for the device's functioning, and weather conditions of the predefined area. As a predefined area of interest, we decided to select Leuven, as it is the city where the DingNet system predominantly works.

Our strategy implies that once a month we calculate the aging coefficient and apply it to the current energy consumption value. This coefficient consists of 3 components:

1. Base calculation according to the aging function described in 2.4.1;
2. Weather factor described in 2.4.2;
3. Usage factor described in 2.4.3.

2.4.1. Base aging model

Determination of the base aging function was tough due to a big variety of options. A lot of studies were conducted in order to find out the model for the process of aging of the device [5], [6], [7]. However, the approaches were different and the resulting model sometimes differed from each other significantly. For example, only in [5] 5 different models were considered in detail. We also needed to adapt those models for the energy consumption calculation. We investigated the materials and found the model which is the most commonly used in device aging research.

According to [5], the Arrhenius model relates the variables of time and temperature to the deterioration of materials and therefore is useful in calculating accelerated thermal aging parameters. It was used for further studies and in [8] authors performed a comprehensive comparison study of different models on the example of refrigerators.

For the model for the current study, we decided to come up with the following approach – as all the previous works are conducted in different fields and experiments are set on different devices types, none of which perfectly match our case, we build our model based on the averaging the set of papers with similar results and devices types.

The delta energy consumption function in the course of aging is the following:

$$E(x) = 1 + 0.595 * (1 - e^{-(\frac{x}{6.517})}) \qquad (1)$$

$$E(x) > 1, x > 0,$$

where *E(x)* refers to the **delta** in energy consumption in comparison to the initial consumption of the device, and *x* refers to time passed since the initial age. The plot for this function can be observed in Figure X.

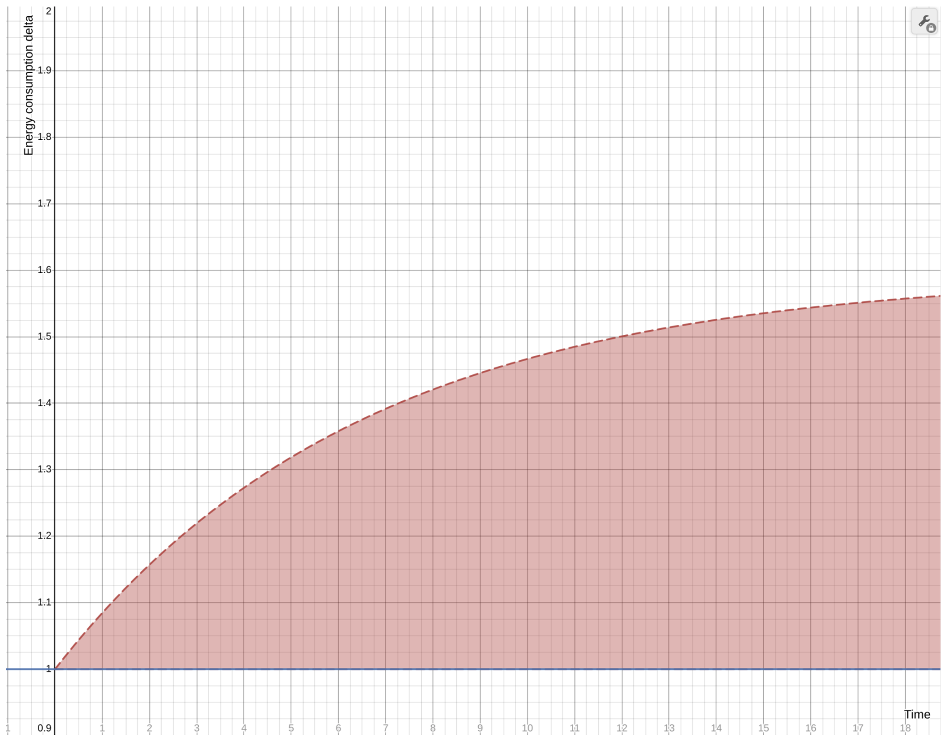


Figure 4. Energy consumption delta in the course of the device lifetime

As we need to compensate for the energy consumption delta which is growing in the course of the lifetime of the device, we will take the following equation as the base function:

$$\mathrm{I(x)} = E(x) - E(x - 1), \qquad (2)$$

Which denotes to the base amount of the energy we need to add to the device at the time x.
In the following subsections, we will extend this function with the weather and usage coefficients.

2.4.2. Average weather year-round in Leuven

According to the [x] the temperature in Leuven is fluctuating between 0°C and 25°C in the course of the year. We can split the year into 4 groups according to the average temperature:

1. **0°C - 10°C** as cold, for Jan-Mar and Oct-Dec
2. **10°C - 15°C** as chill, for Mar-May and Sep
3. **15°C - 20 °C** as warm, for May-Jun and Aug-Sep
4. **20°C - 25 °C** as hot, for July-Aug

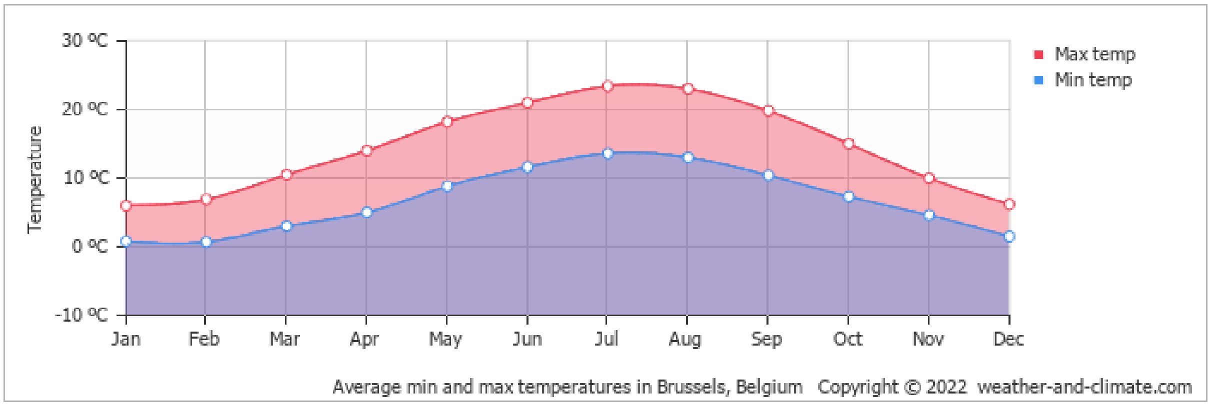


Figure 5. Average weather in Leuven

The fluctuations don't seem very significant for the devices, however, the following research [9], [10] has shown that even this change within 25 degrees can have an effect on the device aging. Therefore, we will enhance our initial model, described in section 2.4.1 by adding the weather factor to the calculation. As we are going to re-evaluate the energy consumption every month, we will add the factor representing the weather group of the target period of the year. We have not found any resources performing the precise mathematical calculation of the weather impact on aging, but based on the study conducted by [10] we decided to define the approximate values for this factor.

$$W_i = 1.05,\ i = 1$$
$$W_i = 1.01,\ i = 2$$
$$W_i = 1,\ i = 3$$
$$W_i = 1.05,\ i = 4$$

where *i* is the group identifier.

The coefficients were introduced on the basis of the assumption that we consider 20 °C a condition with no additional effect on energy consumption [9]. At low internal resistance of the battery increases and it won't be able to deliver the same level of power. It can deliver the current but at a lower voltage level, which will result in lower efficiency of the electronics and therefore higher energy consumption. At the same time, at high temperatures, the internal resistance is lower and this fact increases the battery's ability to deliver energy. This results in faster discharge and a corresponding loss of battery life.

With this coefficient our base formula can be extended in the following way:

$$I'(x) = W(x) * I(x), \qquad (3)$$

where *W(x)* is the function to determine the coefficient based on the weather group of timestamp *x* and *I(x)* is the amount of the energy taken from (2).

2.4.3. Usage factor

The last component for the aging coefficient is the level of the device usage calculated by its total energy consumption.
We take the parameter *C(x)* from the device, where x is the target timestamp and *C(x)* is the total energy consumption from the beginning of the working period of the device.
We conducted a series of experiments on the existing DingNet simulation to determine the difference between the total energy of the same device in the moving and fixed states. It was possible to set the speed for each device and time of the experiment and turned out that if the device moves with a speed higher than 10m/sec, the package loss would be 99%, which is totally not acceptable. There is also a direct dependency on the time and energy consumption from the length of the path. We obtained a sample of the 30 experiments, which included:

- Total energy consumption for the device in the fixed state
- Total energy consumption for the device in the moving state
- Total energy consumption for the device in the fixed state with the high speed (5m/sec<)

Based on this sample we obtained the following results:

1. Total energy consumption does not depend on the speed of the device;
2. Total energy consumption does depend on the distance length;
3. The average total consumption for the track of 300 m was 400 MJ, 800 m - 805 MJ;
4. The total consumption of the fixed device is 0.

Based on the results mentioned above, to compensate for the difference in energy consumption of the fixed and non-fixed device, we introduce a usage coefficient *M(x)*:

$$M_1 = 1.0,\ M_2 = 1.05,$$

where M_1 is a coefficient for the fixed state, and M_2 is the coefficient for the moving state.
This way the formula (3) can be enhanced:

$$I''(x) = M(x) * I'(x) , \qquad (4)$$

Where *M(x)* is a function to determine the usage coefficient of the device. This way the final formula is the following:

$$I''(x) = M(x) * W(x) * (E(x) - E(x - 1)) , \qquad (6)$$
$$E(x) = 1 + 0.595 * (1 - e^{-(\frac{x}{6.517})}),$$
$$x > 0,$$

where *x* is the time stamp of the interest.

2.5. Design of the extension

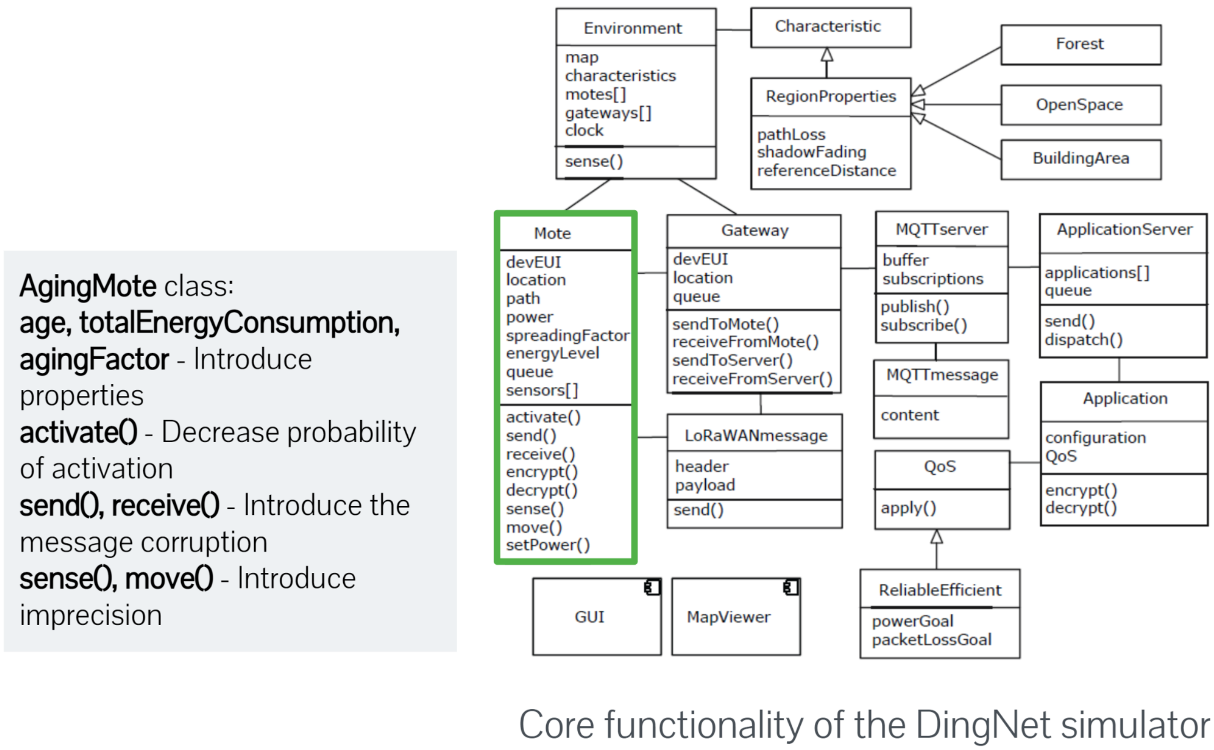
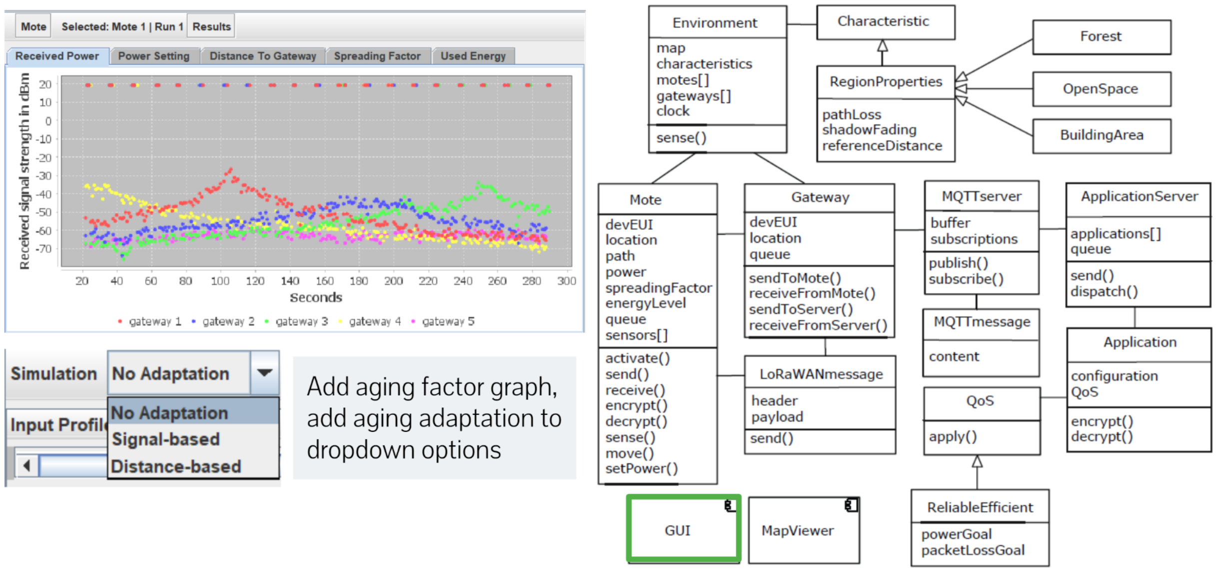


Figure 6. Mote entity extension

The “Mote” class should be extended with an “AgingMote” class by introducing new “age”, “totalEnergyConsumption” properties and an “agingFactor” field. Based on the aging factor the behaviors of existing methods should be modified to degrade the performance of older Motes. For methods “send” and “receive” the probability of message corruption should be introduced. Sensors and actions taken by motes should also become imprecise by modifying the “sense” and “move” methods. Finally, the age factor should have an impact on the mote's probability of activation at the beginning of the simulation, resulting in a lower probability for older motes.



Core functionality of the DingNet simulator

Figure 7. Graphical user interface extension

“GUI” (Graphical User Interface) should be extended to add a new adaptation strategy for the simulation.

Besides that, in order to understand the percentage of a mote’s degradation, the aging factor should be depicted on a graph.

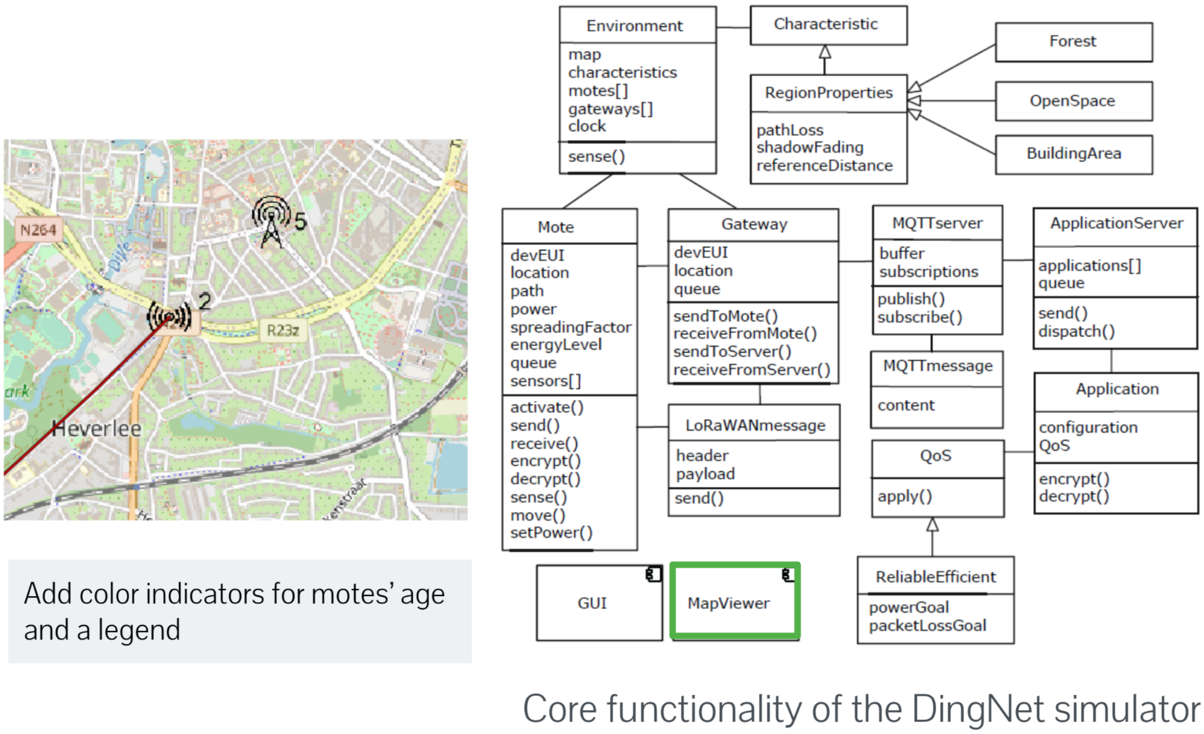


Figure 8. graphical user interface extension - Map

"MapView" should be extended to visually show the distinction between young and old motes. That can be done by introducing a color for mote icons and adding a legend. "MapView" technically is part of "GUI", however in the provided class diagram "Core functionality of the DingNet simulator" they are separated explicitly.

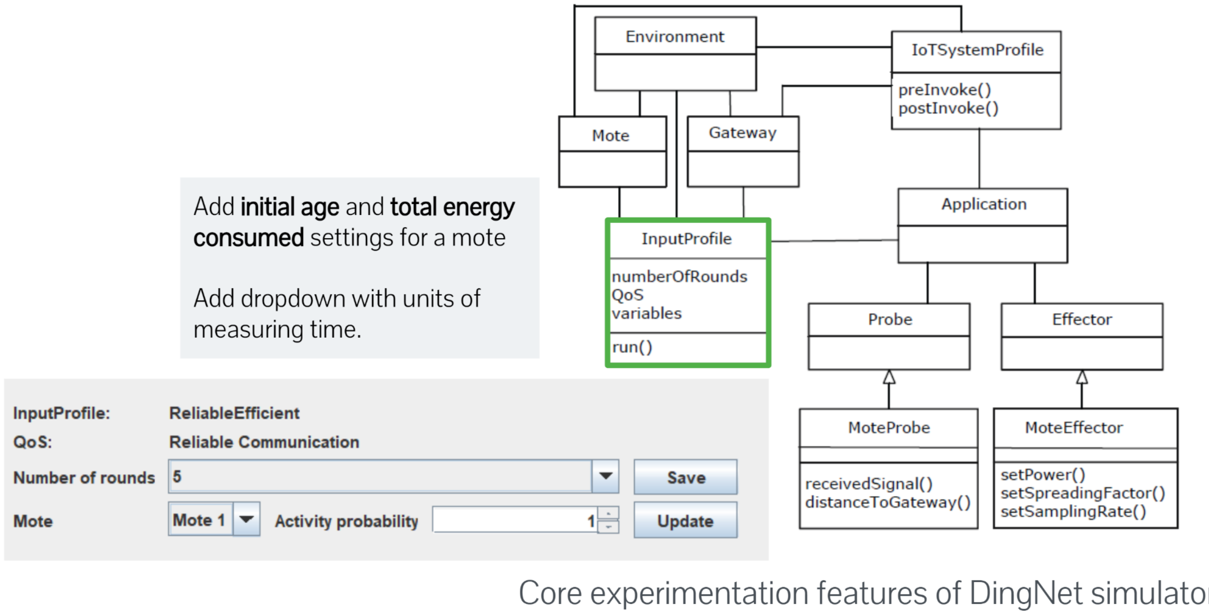


Figure 9. Simulation input profile extension

Each mote can be configured before starting the simulation via “InputProfile” variables. In order to apply a new adaptation strategy, more variables should be defined for every mote, for instance, its initial age, the starting date of a simulation, and its total consumed power.

Another extension would be to introduce a dropdown with units of measuring time for a user to choose. At the moment, simulation only supports milliseconds as its default time setting. Longer simulation periods are needed to illustrate the effects of aging on motes, so options for weeks, months, or years should be available.

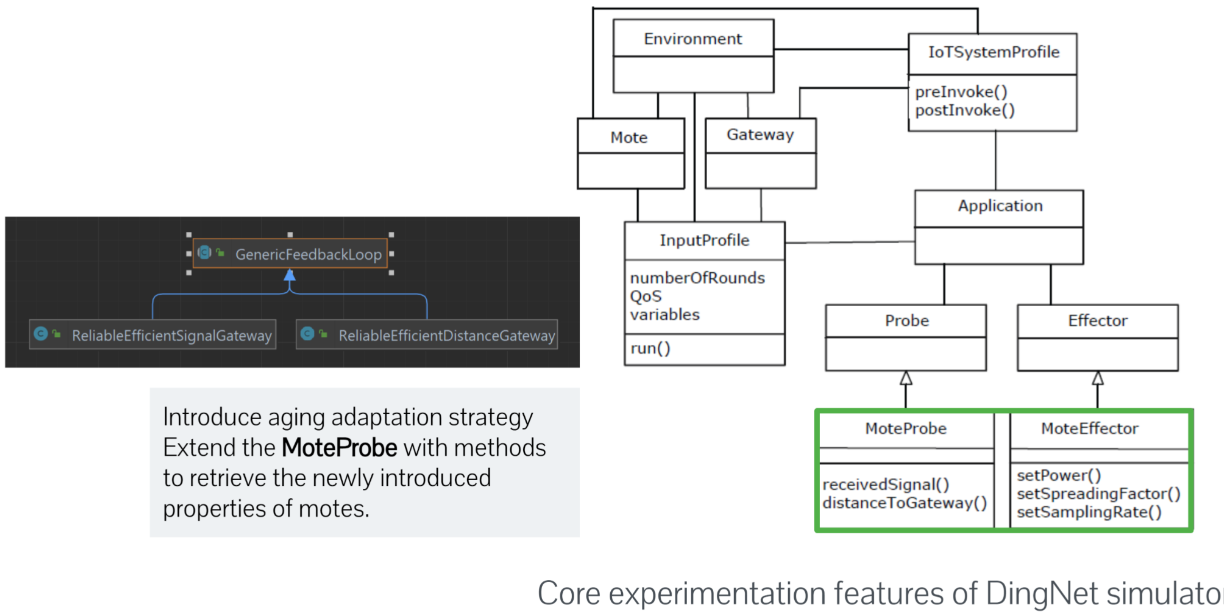


Figure 10. Feedback loop mechanism extension

There is a "GenericFeedbackLoop" abstract class that represents a generic adaptation strategy. It defines adaptation goals and communicates with both "MoteProbe" and "MoteEffector" in order to adjust the managed system. Since there are already two adaptation strategies in place, a third aging adaptation strategy ought to be added.

"MoteProbe" should also be extended in order to make the newly introduced properties accessible to the aging adaptation strategy.

3. Implementation and demo

3.1. Overall description of final design

The goal of this implementation is to improve the reliability of devices and reduce ambiguity and imprecision by addressing the uncertainty of aging devices. Failing to address this uncertainty can lead to package loss and reduced trustworthiness of simulations. The proposed solution involves adapting the system to account for aging devices in order to maintain reliable function.

Implementing the adaptation will allow us to handle the defined uncertainty, and lead to improved accuracy of experiments and more accurate data representation in plots, compared to a non-SAS approach. Packet loss and energy consumption can be used as metrics to evaluate the effectiveness of the adaptation. The SAS approach may result in lower average packet loss on the device, but it may also have the potential drawback of higher motes' energy consumption compared to a non-SAS approach. The MAPE-K loop of the SAS is shown in Figure 11.

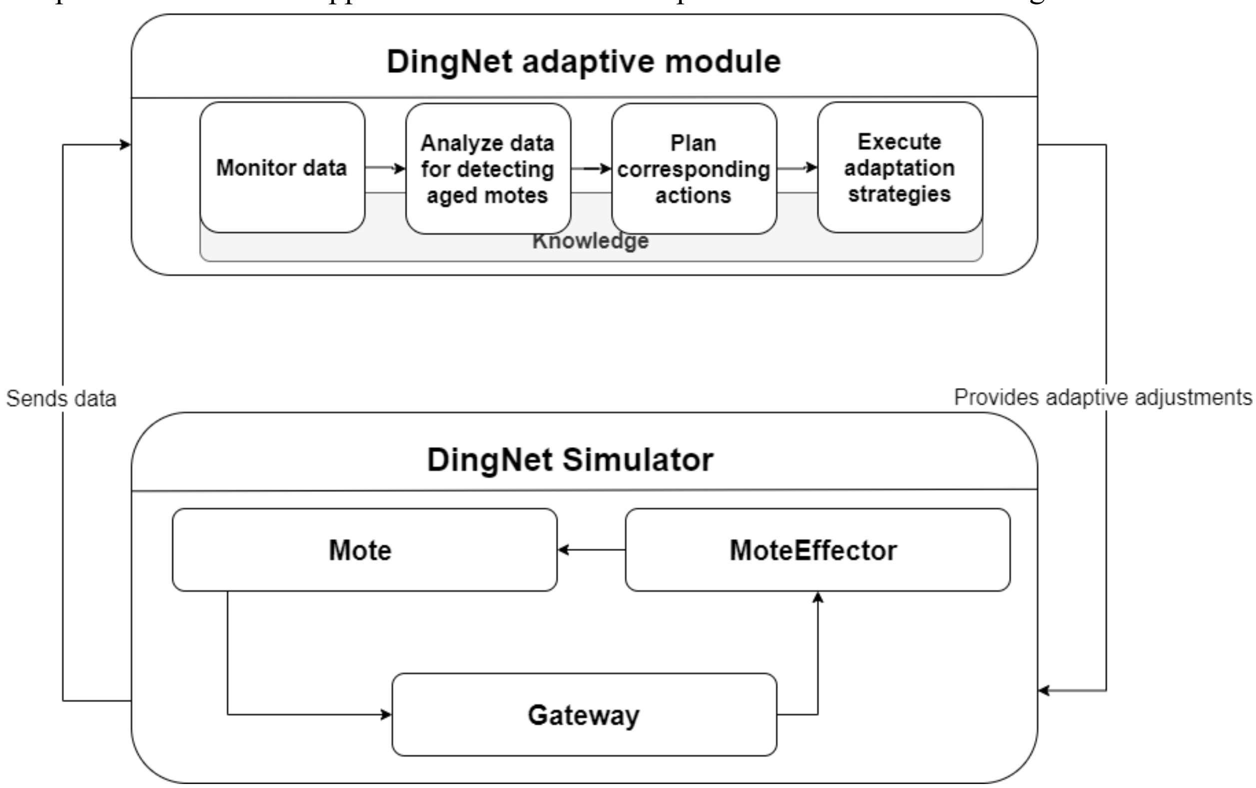


Figure 11: MAPE-K loop for DingNet Extension

The goal of the proposed extension was to meet the following functional requirement of adaptively adjusting energy values to compensate for the deteriorating performance of the device, considering its age.

The system has the following RELAX requirements:

Operation

O1’: The system SHALL ensure that the package loss rate is AS CLOSE AS POSSIBLE TO zero during the operation time. The system SHALL increase the power supply, based on the age of the device AS EARLY as it starts to affect the characteristics of the device. MONITOR: Device aging coefficient, Device energy monitors. ENVIRONMENT: Package loss rate.

O2’: The system SHALL consume AS FEW units of energy AS POSSIBLE during normal operation. MONITOR: Device energy monitors. ENVIRONMENT: Total energy consumption.

Aging is a significant reliability threat, particularly in safety-critical applications like the automotive industry. It causes the performance and eventual failure of digital circuits to degrade over time. The proposed solution aims to address the initial age, total energy consumption, and weather conditions of the predefined area (Leuven, where the DingNet system predominantly operates) as major factors. The strategy involves calculating and applying an aging coefficient to the current energy consumption value on a monthly basis. This coefficient is composed of three components: a base calculation based on an aging function, a weather factor, and a usage factor.

The "Mote" class is suggested to be extended with an "AgingMote" class that includes new properties for age and total energy consumption, as well as an aging factor field. The behaviors of existing methods in the "Mote" class were modified based on the aging factor to degrade the performance of older motes. This includes introducing a probability of message corruption for the "send" and "receive" methods, making sensor readings and actions taken by motes imprecise by modifying the "sense" and "move" methods, and decreasing the probability of activation for older motes at the beginning of the simulation.

Figure 12. Mote class extension

The graphical user interface (GUI) should be extended to add a new adaptation strategy for the simulation and display the aging factor on a graph to understand the percentage of a mote's degradation. The "MapView" feature, which is technically part of the GUI but is shown separately in the provided class diagram, should be extended to visually distinguish between young and old motes by using a gradient color from green to red signifying the agingFactor. The color has a mid value of yellow when having a tolerant agingFactor.

In order to implement a new adaptation strategy, more variables such as initial age, starting date, and total consumed power should be defined for each mote in the "InputProfile" configuration before starting the simulation. The GUI should also include a dropdown with units of time for the user to choose, as the current simulation only supports milliseconds as the default time setting. To illustrate the effects of aging on motes, options for longer periods such as weeks, months, or years should be available. The "GenericFeedbackLoop" abstract class, which represents a generic adaptation strategy and communicates with the "MoteProbe" and "MoteEffector" to adjust the managed system, should be extended to include a third aging adaptation strategy. The "MoteProbe" class should also be extended to make the newly introduced properties accessible to the aging adaptation strategy.

3.2. Implementation

Technologies:
The software including the extension is written in Java. There are several components used by the infrastructure:

- MQTT[11] Server
- Application Server
- Simulation of nodes and sensors
- Antenna/Gateway Network server

MQTT technology is a lightweight publish/subscribe messaging protocol in low bandwidth environments. The use of the MQTT protocol allows to establish communication between devices, with low power and bandwidth consumption. MQTT in the DingNet project is used to send messages from motes to the server.

The following libraries were used in this project:

- Jfreechart - provides the visualization of charts
- Jxmapviewer - provides the visualization of the map and some other dependent interface features

Detailed design:
During the work, the existing classes were expanded, and new elements were added to the architectural component of the project. The features of the project are shown in Figure 13.

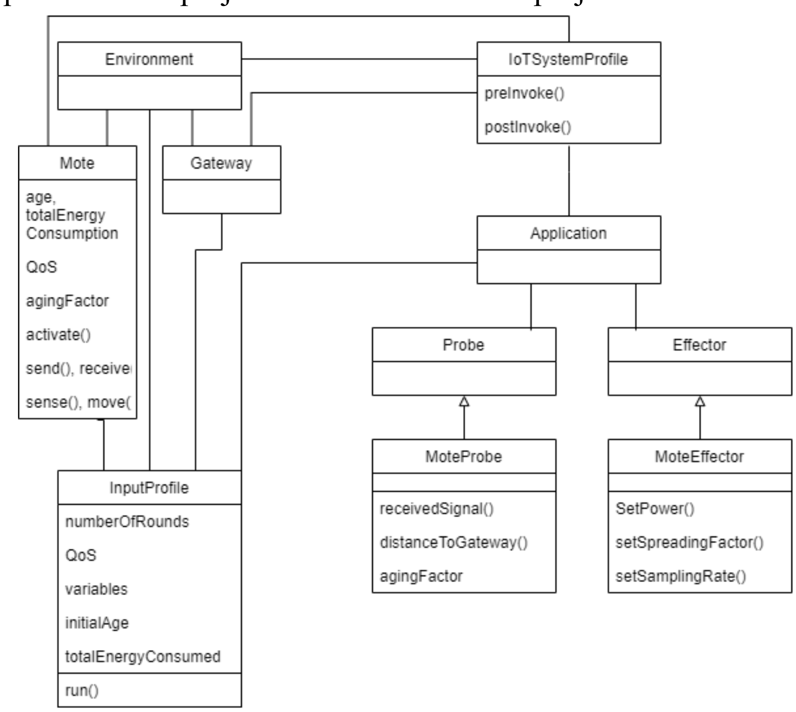


Figure 13: Features of DingNet simulator with added extension

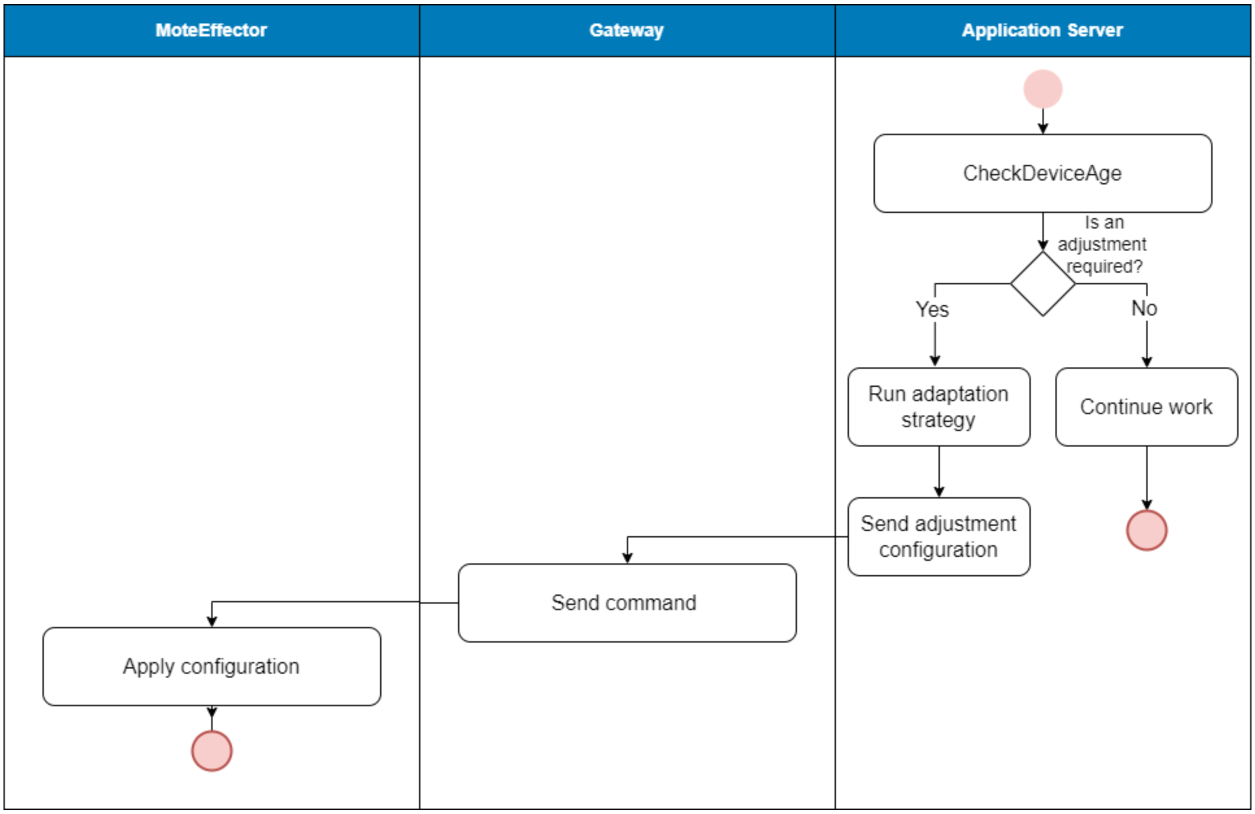


Figure 14: Activity diagram of the extension

Input Profile configuration:
The edited window of the input profile is shown in Figure 15.
There are several new groups of elements added:

- Device's lifespan shows the expected device's service life expiration time
- Device adjustment rate - every month.
- Time measurement - different time parameters configurable by the user .
- Mote settings - represent different parameters of individual mote

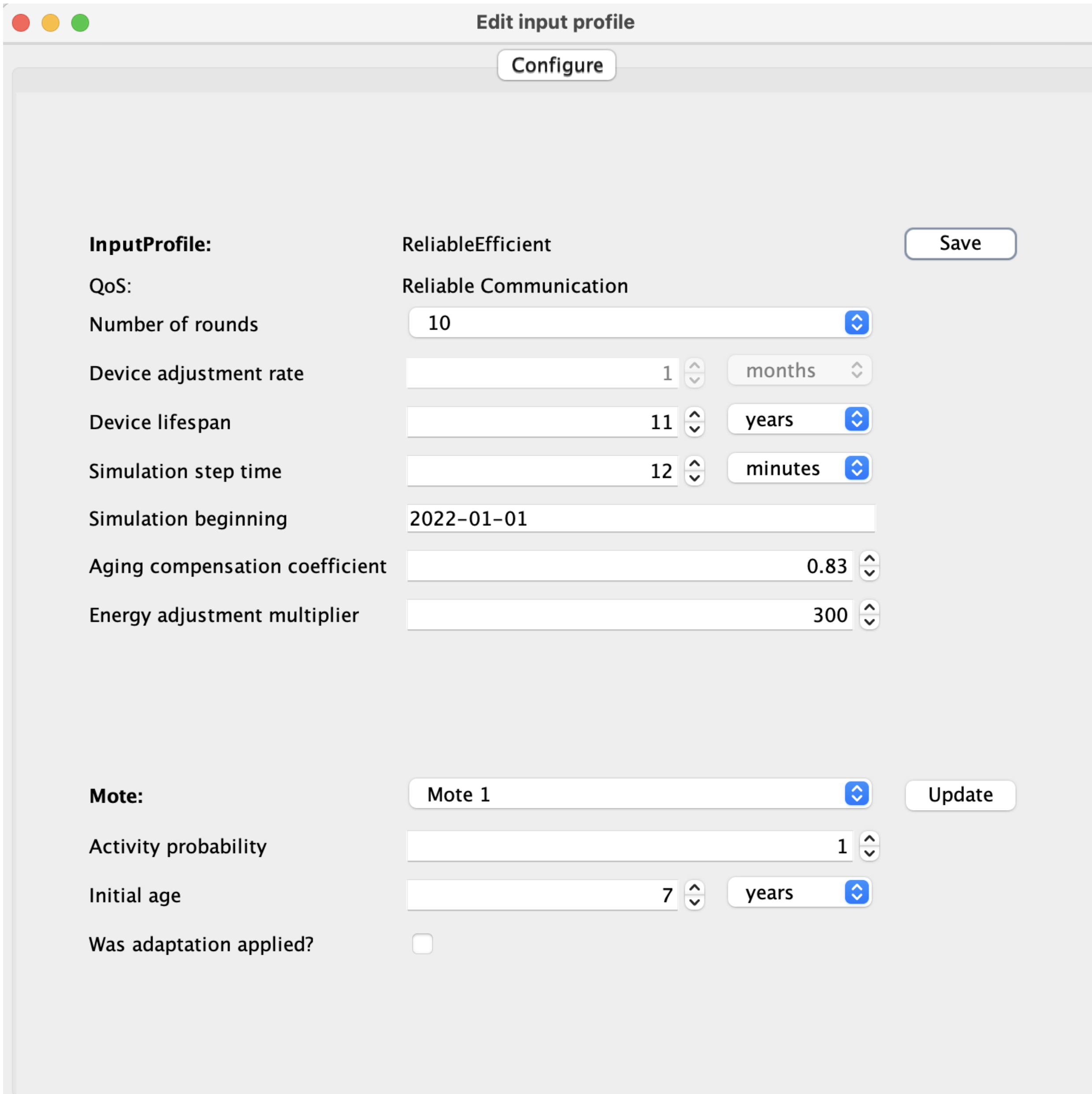


Figure 15: Input profile view

Description of aging mote:

1. Each mote is being initialized with initial age and total Energy Consumed. Device's lifespan can't be 0 if total Energy Consumed > 0). It is assumed that total energy consumed includes also all the adjustments made to compensate for the aging (if device age is equal or more than 1 month)
2. Let assume that aging was handled properly throughout the device's lifespan, therefore knowing the compensation rate, we can calculate initial aging factor as follows:
   initial age / device's lifespan * (1 - compensation coefficient)
3. Each mote sends current age to gateway

Aging factor - used as probability to degrade devices. Ranges [0..1].
Every time device's age is updated, aging factor also updates by aging factor unit = *time measurement(sec) / device's lifespan (sec) * compensation coefficient*

Devices without any adjustments completely fail after 10 years.
When energy delta is calculated (inside Aging Calculator), we also calculate the aging factor adjustment, which is *device adjustment rate / time measurement * aging factor unit * compensation coefficient.*

Algorithms:
It was required to re-implement the date-time formats provided in the original project, as Lora Transmission records only the time, and not the date.
Simulation time is being tracked in nanoseconds, therefore it is possible to convert the time to appropriate format, which is required for adaptation strategy.
Duration.ofDays(365*15).toNanos()
Convert simulation time to total ticks.
this.environment.clock.toNanoOfDay()/LocalTime.NANOS_PER_MILLI = 378_652
Duration.ofDays(1).toSeconds() = 86_400
Duration.ofDays(1).toMillis() = Duration.ofDays(1).toSeconds()*LocalTime.MILLIS_PER_SECOND = 86_400_000
Duration.ofDays(365*20).toMillis() = 630_720_000_000
Every tick Duration.ofMillis(1).toNanos() OR ChronoUnit.MILLIS.getDuration().toNanos() OR LocalTime.NANOS_PER_MILLI = 1_ 000_000 is added to the simulation timer.
Simulation considers duration to be in seconds, so it's required to do conversion:.
LocalTimer.MillisPerDay etc.
LocalTime.MILLIS_PER_SECOND
The following formula was used to count the age of the mote:

$$E(x) = 1 + 0.595 * (1 - e^{-(\frac{x}{6.517})})$$

Alternative implementation paths:
During the analysis, various implementation options were considered. Among them were implementation options of both software and physical nature taking into account the use of the simulator in real conditions. These options are discussed in Table 7.

Concern (Identifier: Description)		*Con#1: How should information about the age of a mote be stored?*
Ranking criteria (Identifier: Name)		*Cr#1: Resource efficiency* *Cr#2: Complexity of implementation*
Options	***Identifier: Name***	*Con#1-Opt#1: In mote itself*
	Description	*Age is stored in the mote, calculated and sent at a periodic time interval*
	Status	*This option is rejected.*
	Relationship(s)	-
	Evaluation	*Cr#1: Sending age status regularly leads to redundancy, and increased bandwidth usage*

		Cr#2: Sending an age would require adding a new MQTT channel, which is simple to create but adds additional complexity to the system. In addition, this adds difficulties when replacing parts of the device, since in order to change the age value, it will be required to send a new configuration to the device itself.
	Rationale of decision	*This option is rejected because it adds unnecessary complexity to the system, besides the need to store the value on the mote and constantly send the age to the server is not suitable for logical reasons.*
	Identifier: Name	*Con#1-Opt#2: In the configuration of simulation*
	Description	*Age is stored in configuration of the application server*
	Status	*This option is rejected.*
	Relationship(s)	-
	Evaluation	*Cr#1: This option implies optimal energy consumption and resource consumption* *Cr#2: Storing the parameter in the configuration file imposes some restrictions, in particular, the dynamic change of the parameter*
	Rationale of decision	*This option is rejected, due to the imposed restrictions in the implementation of the solution.*
	Identifier: Name	*Con#1-Opt#3: In the profile of simulation*
	Description	*Age is stored in profile of the application server*
	Status	*This option is decided.*
	Relationship(s)	-
	Evaluation	*Cr#1: This option implies optimal energy consumption and resource consumption* *Cr#2: Storing the parameter in the profile allows to implement all the features specified in the extension requirements*
	Rationale of decision	*This option is selected because it is both resource efficient and allows the task to be completed in accordance with logical considerations and requirements*

Table 7: Description and rationale for choosing a solution

Implementation process:

During the development of the software part and writing the accompanying documentation, it was decided to use a modified Agile methodology, due to the small amount of time and the impossibility of using full-fledged sprints. 4 weekly sprints with regular stand-ups were chosen, where the achieved results, blockers and possible tasks for the next sprint were discussed. Thus, it turned out to distribute tasks among team members, set priorities and set a time frame for when one or another component should be ready.

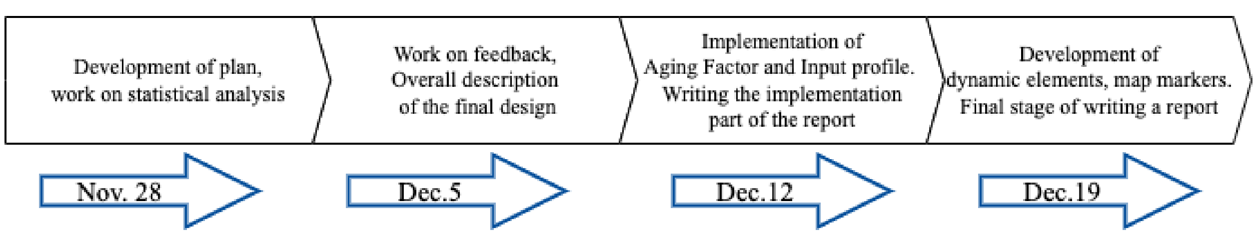


Figure 16: Timeline of the project

3.3. Showcase and evaluation

Demonstration:

A demonstration of the extension is shown in the following video: [Link to the video](https://drive.google.com/file/d/1TmOKzNW0kdCn2ol5-RK4HSD_Ewxo2wvu/view?usp=sharinga)
(https://drive.google.com/file/d/1TmOKzNW0kdCn2ol5-RK4HSD_Ewxo2wvu/view?usp=sharinga)

It is divided into several parts:

- 0:00 - Comparison with original DingNet project
- 1:15 - Usage of DingNet extension
- 1:40 - Input profile explanation
- 2:29 - Results of the extension

Advantages of the system:

This adaptive strategy is suitable for all categories of devices, allowing it to receive correct data, despite the age of the device and the decrease in signal transmission characteristics.

Among the main advantages of the extension are:

- Ability to receive data from aging nodes - by adjusting power settings, correct data can be received.
- Extending the life of motes - the usage period of mote is longer, without serious deterioration in quality.
- Motes lifetime monitoring - we can monitor lifetime of motes, and determine the age of the device.
- Visualization of broken motes - broken and incapable motes are visualized on the map.

Limitations:

Despite all the advantages, the extension is also subject to some limitations, which are however open to be resolved in future releases. Among them is the fact that we can only set monthly device adjustment rate, as weather was classified by months.

Quantitative evaluation:

As a result of developing an adaptive strategy, and conducting multiple tests, the following results were recorded:

When using the strategy on an older device:

- It increases the life of the device,

- Reduces packet loss, depending on the age of the device,
- But at the same time increases the amount of energy consumed.

The results of the tests can be seen in Figure 17.

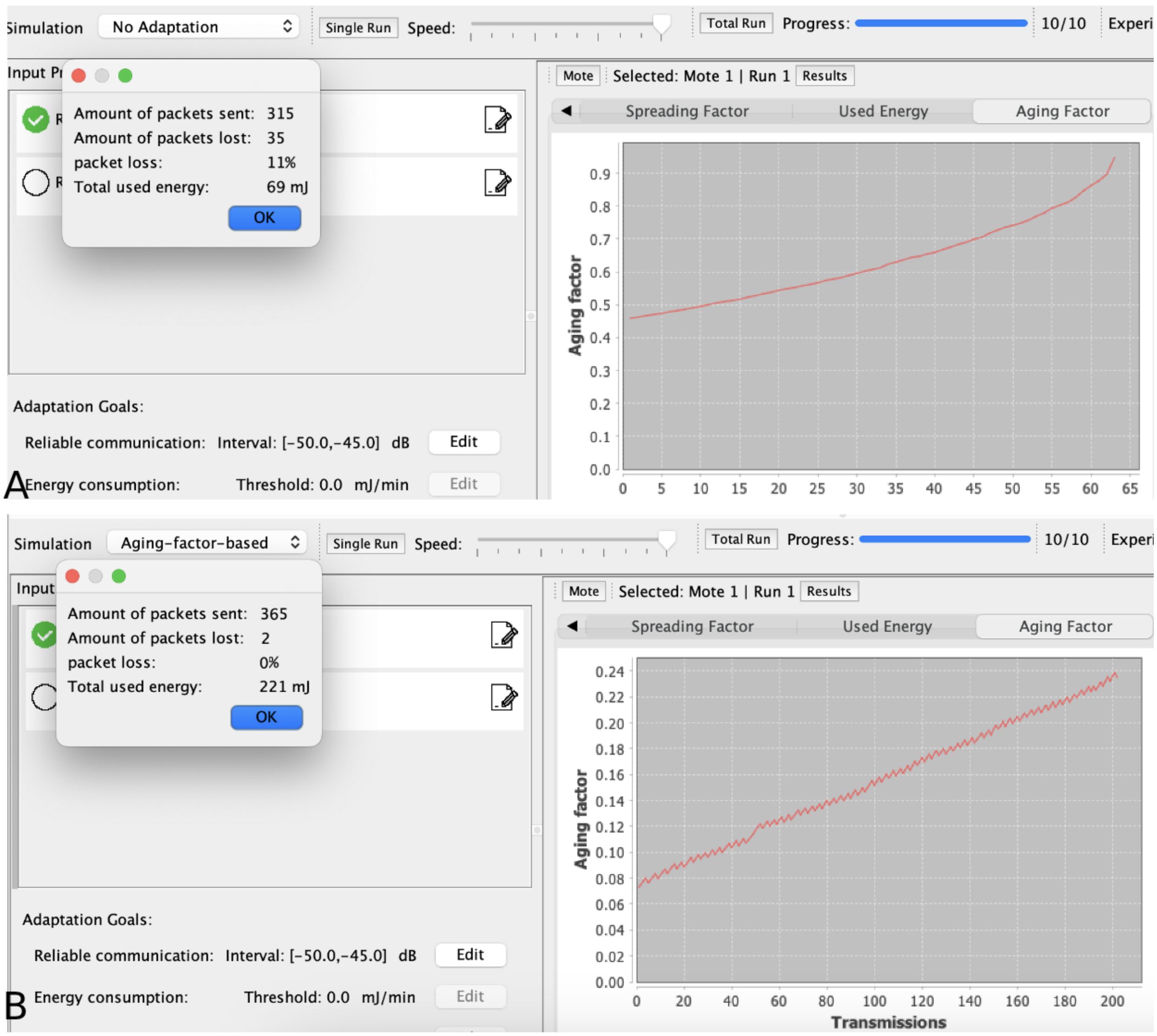


Figure 17: Comparison of indicators when working without an adaptation strategy (A) and with an adaptation strategy (B)

3.4. Reflection

Takeaways:

This chapter describes the main take-aways that we managed to take out as a result of the development of the project and accompanying documentation:

- Development of adaptive software
- Development of managing and managed systems
- Working with adaptation loops
- Different approaches of adaptive software

Challenges:

The main challenge was the source code of the original project. Since it was not written in accordance with best practices .It took time to analyze and implement features.

The problems we faced with the code:

- Classes with 2000+ lines
- Code duplications
- Incorrectly chosen data structures
- Hardly any software design principles that were followed
- Inconsistent and incorrect time units used throughout the simulation
- Mutable domain objects that had multiple sources of control which were constantly mutating them
- High coupling and low cohesion
- Incorrect build and source control configuration. Build did not work from scratch, only when old artifacts exist

Also there are a couple of limitations that can be seen in the project:

- Lora Transmission records only the time, and not the date. As we want to extend it, we had to re-implement the date-time formats provided.
- Some parts of the original project needed to be rewritten due to incorrect logic, or a non-working solution.

Split of the work and coordination:

Regular offline meetings were allocated to work on the project. Weekly meetings were held to set goals and discuss the results achieved. In case of need or adjustment of actions, online meetings were organized. The distribution of work among group members is shown in Table 8.

Alina Boshchenko	Analysis of uncertainties, analysis of potential solutions parts of the report; adaptation concept and description, devices aging literature research, code analysis
Dmitriy Knyajev	Analysis of need for this uncertainty, Requirements, Implementation, Showcase and

	evaluation, reflections and experiment reproduction parts of the report
Nichita Goncear	Expected effects of using the extension, overall system description, final design description, dynamic colors of motes on the map
Dmitrijs Voronovs	Uncertainty and adaptation idea, description of the extension, solution analysis, work coordination, repository setup, code and existing implementation analysis, implementation of extension and adaptation strategy, extension documentation

Table 8: The distribution of work

3.5. GitHub Project and Experiment Reproduction

Link to GitHub repository: https://github.com/dmitrijs-voronovs/dingNet/tree/extension
As a result of the work:

- 226 files were changed,
- 2460 lines of code added.

In order to repeat the experiment and use the extension the following components are required:

- IntelliJ IDEA
- JDK ver. 17 or higher
- DingNet files from repository

To run the project:

1. Open the DingNet folder in IntelliJ IDEA.
2. Run the main method of the class MapView: Press the "Run" button at the top right corner.
3. When DingNet main windows is open choose Configuration file.
4. Select required adaptation strategy from the list of adaptation strategies.
5. Select profile.
6. Run the simulation.
7. The results of the adaptation will appear on the screen.

References

[1] M. T. H. Anik, S. Guilley, J. -L. Danger and N. Karimi, "On the Effect of Aging on Digital Sensors," *2020 33rd International Conference on VLSI Design and 2020 19th International Conference on Embedded Systems (VLSID)*, 2020, pp. 189-194, doi: 10.1109/VLSID49098.2020.00050.
[2] M. Provoost and D. Weyns, "DingNet: A Self-Adaptive Internet-of-Things Exemplar," 2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2019, pp. 195-201, doi: 10.1109/SEAMS.2019.00033.
[3] *Whittle, J., Sawyer, P., Bencomo, N. et al. RELAX: a language to address uncertainty in self-adaptive systems requirement. Requirements Eng 15, 177–196 (2010), doi: 0.1007/s00766-010-0101-0*
[4] A. S. MUTSCHLER, Dealing With Device Aging At Advanced Nodes, SemiEngineering, 2020 https://semiengineering.com/dealing-with-device-aging-at-advanced-nodes/
[5] S.P Carfagno, R.J. Gibson, A review ofEquipment Aging Theory and Technology, 1980, https://www.osti.gov/servlets/purl/5032834
[6] Rodríguez, A., Pérez-Artieda, G., Beisti, I. et al. Influence of Temperature and Aging on the Thermal Contact Resistance in Thermoelectric Devices. J. Electron. Mater. 49, 2943–2953 (2020). https://doi.org/10.1007/s11664-020-08015-y
[7] D. Kraak *et al*., "Device aging: A reliability and security concern," *2018 IEEE 23rd European Test Symposium (ETS)*, 2018, pp. 1-10, doi: 10.1109/ETS.2018.8400702.
[8] Paul, Andreas & Baumhögger, Elmar & Elsner, Andreas & Reineke, Michael & Hueppe, Christian & Stamminger, Rainer & Hoelscher, Heike & Wagner, Hendrik & Gries, Ulrich & Becker, Wolfgang & Vrabec, Jadran. (2021). Impact of aging on the energy efficiency of household refrigerating appliances. Applied Thermal Engineering. 205. 117992. 10.1016/j.applthermaleng.2021.117992.
[9] SaftBatteries, The impact of the temperature on your IoT application's power consumption, 2021, https://www.saftbatteries.com/energizing-iot/impact-temperature-your-iot-application%E2%80%99s-power-consumption
[10] R. Narang, M. Saxena, R. S. Gupta and M. Gupta, "Impact of Temperature Variations on the Device and Circuit Performance of Tunnel FET: A Simulation Study," in IEEE Transactions on Nanotechnology, vol. 12, no. 6, pp. 951-957, Nov. 2013, doi: 10.1109/TNANO.2013.2276401.
[11] B. Mishra and A. Kertesz, "The Use of MQTT in M2M and IoT Systems: A Survey," in IEEE Access, vol. 8, pp. 201071-201086, 2020, doi: 10.1109/ACCESS.2020.3035849.