Final project report

MPSE2201 System design and engineering

Group 10

April 12, 2020

Abstract

Final project report in Systems Design and Engineering. The report explains how we would design a system to make large apartment complexes reach a zero-emission goal and how to employ a system engineering approach to develop the product.

Contents

Abstract ii

Contents iii

List of Figures viii

Abbreviations xi

Introduction 1

Assignment . . . 1

Report . . . 1

Our Systems Engineering Process . . . 2

Planning 3

Contract of Operations . . . 3

Complications due to COVID-19 . . . 5

Development risk analysis . . . 5

Tools . . . 6

Internal Discussions . . . 12

The Problem 14

Understanding the Problem . . . 14

Context . . . 15

Stakeholders . . . 18

Requirements 20

User Requirements . . . 20

Stakeholder concerns . . . 20

Analysis of Competition . . . 21
Effectiveness Measures . . . 22
System Requirements . . . 23
Energy Consumption . . . 24

System Architecting **25**

Tier 0: The System in Its Context . . . 26
Operation . . . 26
Behavior . . . 28
Structure . . . 28
Harvesting Energy . . . 30
Storing Energy . . . 31
Generating Heat . . . 32
Trade-Off . . . 33
Harvesting Energy . . . 33
Storing Energy . . . 34
Generating Heat . . . 36
Recommendations . . . 36
Derived Requirements . . . 37
Tier 1: The System . . . 37
Operation . . . 37
Control System . . . 39
Photovoltaic System . . . 40
Energy Storage system . . . 40
Geothermal Heating/Cooling . . . 41
Behavior . . . 41

Structure . . . 43
Trade-Off . . . 44
Control System . . . 44
Recommendations . . . 45
Derived Requirements . . . 45
Tier 2: Subsystems . . . 47
Energy Storage System . . . 47
Behavior . . . 47
Structure . . . 48
Control System . . . 49
Operation . . . 49
Behavior . . . 49
Structure . . . 55

Build and Test Plan **56**

Verification . . . 56
Validation . . . 57
Prospective Validation . . . 57
Retrospective Validation . . . 57

Verification **57**

Production . . . 57
Capacity for Storage . . . 59
SOLIDWORKS MODELL Temperatur FINN PÅ BEDRE TITTEL . . . 60
SOLIDWORK Solar CElls weight FINN PÅ BEDRE TITTEL . . . 60
Prototypes . . . 60

Machine Learning . . . 60
Predicting Solar Panel Power Production . . . 60
Predicting Power Prices . . . 62
Web-Application . . . 65

System Risk Analysis **66**

Economics **67**

Our Budget . . . 67
Our Cash Flow . . . 69
Residents Budget . . . 69
Residents Cash Flow . . . 70

Conclusion **70**

About our system . . . 70
What did we learn . . . 71
What could be improved . . . 71

References **73**

Appendices **74**

A Emails . . . 74
A.1 Geothermal heat pump supplier email . . . 74
A.2 Lithium battery supplier email . . . 75
A.3 Solar panel supplier email . . . 76
B Code . . . 77
B.1 Historical Weather Web-Crawler . . . 77
B.2 MongoDB Filler . . . 79

B.3 Historical Hourly Avg. Power Price 83
B.4 Machine Learning 84
Solar Power Production 84
Machine Learning, Power Price Prediction 86
B.5 Web-Application 88
script.js 88
nav.php 92
header.php 92
footer.php 92
Temperatur index.php 93
endre.php 93
index.php 94
TemperaturModel.php 94
TemperaturController.php 95
HomeController.php 96
Application.php 96
Model.php 97
View.php 97
Controller.php 97

List of Figures

1 Risk matrix for development 5

2 Jira screenshot 6

3 Microsoft Teams screenshot 7

4 Gantt diagram screenshot 9

5 Github web-application repo screenshot 11

6 MongoDB screenshot 12

7 The location of our building situated in Oslo 14

8 The initial sketch of the context 16

9 A 3D model of the building 17

10 The man-made systems in our systems immediate context 18

11 An onion diagram showing the stakeholders 19

12 Key-driver-graph for OBOS 22

13 Key-driver-graph for the users of our system 23

14 A technical budget displaying the energy needs of the building 25

15 Use case diagram for the system 26

16 Black box of our system 27

17 FFBD within our system 28

18 The system modeled as a UML class diagram, where all the classes are still abstract 29

19 Morphological diagram, showing possible concepts 30

20 Pugh matrix for harvesting energy 34

21 Pugh matrix for storing energy 35

22 Pugh matrix for generating heat 36

23 Morphological diagram, showing the recommended concept 37

24 Use case diagram, between subsystems 38

25 Black box of Control System . . . 39
26 Black box of Photovoltaic System . . . 40
27 Black box of Energy Storage System . . . 40
28 Black box of Geothermal Heating System . . . 41
29 Black box of Geothermal Cooling System . . . 41
30 FFBD of the respective subsystems . . . 42
31 Class diagram showing the subsystems . . . 43
32 A more detailed structural model of our system and its context . . . 44
33 Pugh matrix for Control System . . . 45
34 Average hourly power prices 2013-2020 (Source code on page 83) . . . 46
35 Behavior of the Energy Storage System . . . 47
36 Class diagram of the Energy Storage System . . . 48
37 Use case diagram within the control system . . . 49
38 FFBD for the Control Unit . . . 50
39 FFBD for the MCU . . . 51
40 FFBD for the continuous data gathering software . . . 53
41 FFBD for the machine learning software . . . 53
42 FFBD for the application/website . . . 54
43 FFBD for the developer tools . . . 55
44 Class diagram for Control System . . . 56
45 Graph of estimated yearly power production and consumption . . . 58
46 Graph of estimated 24 hour peak power production . . . 59
47 Machine learning prediction results for shuffled solar production . . . 61
48 Machine learning prediction results for chronological solar production . . . 62
49 Power prices 2013-2020 in NOK/MWh . . . 63

50 Machine learning prediction results for shuffled power prices 64
51 Machine learning prediction results for chronological power prices 65
52 Screenshots of the website prototype 66
53 Risk matrix for system operation 67
54 Budget for our company 68
55 Cash flow graph for our company 69
56 Budget for customer 69
57 Cash flow graph for customer 70

Abbreviations

COVID-19 Corona Virus Disease 2019

CRUD Create Read Update Delete

FFBD Functional Flow Block Diagram

JSON JavaScript Object Notation

MCU Main Communication Unit

SQL Structured Query Language

UML Unified Modeling Language

V2B Vehicle-to-Building

Introduction

Assignment

People are getting more conscious to the impact we make on the environment, and we need to find ways to reduce our emissions. And reduce the amount of fossil fuel we are consuming.

We can also see that we are using more and more power in our homes, according to Norges vassdrags og energidirektorat the amount of power consumption of homes accounts for 48% [14] of Norway's power consumption, in Norway about 70%[5] of this is used for heating of space and domestic hot water.

If we can find ways to generate power in or around the buildings. And reduce the amount of electrical heating we rely on, we can make a positive impact on the world.

In this report we will describe the details of our system and show the design process of our system up until the building phase.

We will design a smart system to help buildings reduce energy consumption and increase energy production, using these systems engineering activities:

- Plan the system engineering approach, to help with our system engineering activities.
- Identify system functional input requirements and needs from stakeholders that have an impact on the system of interest.
- Identify system functional input requirements and needs from stakeholders that have an impact on the system of interest.
- Identify system boundary and use cases.
- Capture system functions and logical interfaces.
- Identify logical architecture.
- Allocate functions to logical architecture.
- Identify physical architecture.
- Create test, verification and validation plan.

Report

This report will explain our approach to solving the assigned problem using systems engineering. We start by explaining our planning and some tools we chose to aid our engineering work.

Understanding the problem and understanding what our customer cares about is important. We also map out other essential stakeholders and their concern. We then talk about the context in which our system will operate.

To explain our product, we start at a high abstract level, then as we go deeper into subsystems, we explain what design choices were made and why.

We touch on the verification plan for some of the most critical requirements from our requirement document. And talk about the performance verification of our system.

We finish the report with a risk matrix for both development and operation. We explain how we will monetize the product. And how our customer will end up profiting from buying it.

Our Systems Engineering Process

In this report we have tried presenting the process in a linear fashion, but of course in reality it was anything but.

The first iteration of the process was kickstarted by Gerrit in the first few weeks of the course, here we went through the 20 steps that would make up the final homework we submitted to Gerrit. This was helpful as we knew nothing about systems engineering at the time. Gradually however, we would start to own the process more and more.

After the initial exploration we had a frame for further exploration. The main goal onward was to unfold the system and to apply the systems engineering process on the subsystems as well. Along the way these explorations led to us revisiting all the previous steps and reconsidering several aspects of our system. For example, the fact that we had to increase the roof area to living area ratio would never have been found as early as it was if the process had been linear. If this was a real project this would have led to major setbacks both in terms of time and money.

There were also several explorations into technologies and options that led to a dead-end, but it was still valuable to dive into them in order to make sure that we found the most viable concept available to us.

In addition to the material offered by the lecturers we have also relied heavily on two of the books[18][19] recommended by the teacher when learning about and applying the methods of systems engineering.

Planning

Contract of Operations

In order to make the teamwork function properly, one of the first things we agreed upon was the contract of operations. This was important in order to consolidate how the dynamics of the group should function, and to avoid conflicts.

Contract of Operations

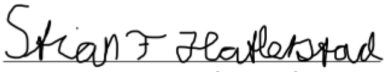
Terra Solaris Borealis

- The team members are obligated to attend all group meetings.
- If a team member is unable to attend a meeting, he shall inform the systems engineer in advance.
- All assigned work shall be submitted before its deadline.
- In case of disagreement, the systems engineer shall call a vote.
- In the case of a tie, the systems engineer has the final say.
- Every opinion and idea have the right to be expressed freely.
- All members of the team shall act with respect towards the others

Tarald Vestbøstad

Simen Hegge

Erik Kvalvik



Stian Hatlestad

Vegard Bernhardsen

Kent Odde

Complications due to COVID-19

After the COVID-19 breakout of 2020, we had to change our plans and "reiterate" our development process in and of itself. Pure online meetings, no more ink and paper. Although we already had connected online using several tools, moving our physical meetings online was an interesting change. In addition to our meetings, our classes were shifted online.

In general, panic and lockdown spread nationwide, and even though we were not smitten with the virus ourselves, all of these factors were indirectly impacting this group.

Development risk analysis

We performed risk analysis for our ongoing development so one of these scenarios do not bankrupt us and cancel the project.

The risk analysis matrix shows how likely each scenario is and how much it will affect us or the stakeholders negatively. And the recommended action is what we can do to reduce the risk or the severity.

RISK	AREAS AFFECTED	SEVERITY	LIKELIHOOD	RISK IMPACT	RECOMMENDED ACTION(S)
Covid-19 pandemic	Developers health	LOW	HIGH	MEDIUM	Home office, and no physical meetings
Project goes over budget and delayed	Investor retention	MEDIUM	LOW	MEDIUM	Use proper system engineering method
Low sales	Developers salary	HIGH	LOW	MEDIUM	Customer surveys, make sure our product is wanted before starting development
Construction/installation delays	Customer satisfaction	MEDIUM	LOW	MEDIUM	Promise economic incentive in the contracts, to finish on time

Figure 1: Risk matrix for development

Tools

Jira

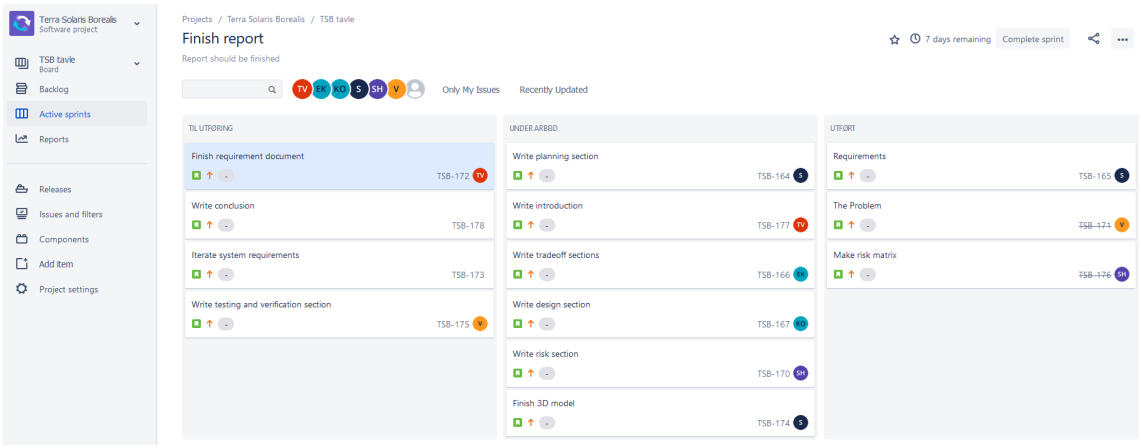


Figure 2: Jira screenshot

Jira is a site/tool used for all types of group work. Originally a software meant to track bugs and issues, reformatted for common use. It has several functions leaning towards iteration and managing projects, through workflows, tracking of issues, progress reports, logging, scrum boards, backlogs, and more, all in all making a good fit for use in our systems engineering project.

Microsoft Teams

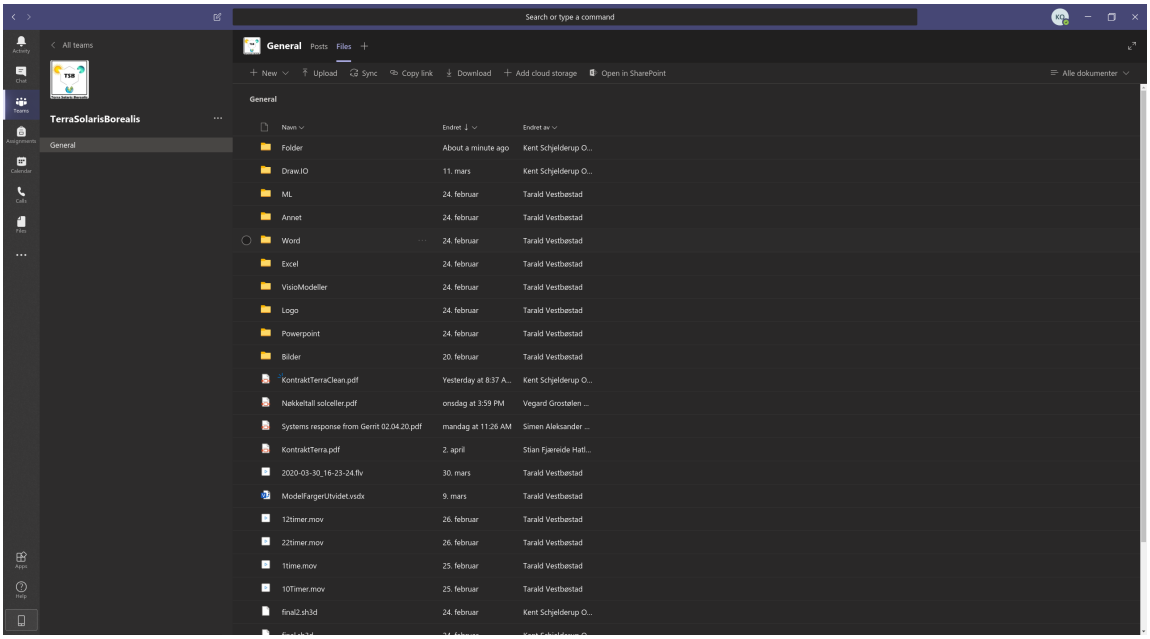


Figure 3: Microsoft Teams screenshot

Teams is a communication platform that includes chat, video conferencing, file storage, file sharing, and application integration. It is the evolution and upgrade path from Microsoft Skype for Business. It allows for efficient collaboration and sharing, keeping us all up to date with the project at all times. Having such a tool is import for our systems engineering project.

Facebook Messenger

Another communication platform that includes chat and calls, more lightweight and easier to communicate using overall. Good for fast messaging and important announcements.

Microsoft Visio

Visio is a diagramming and vector graphics application, capable of mind-mapping, concept design, and flowcharts used in for example systems engineering in general. To our group it was vital in visualizing our project.

Microsoft Word

Word is a commonly used writing tool with a wide variety of uses. When using Teams, Word is an obvious choice for text documents. As it allows multiple people to work on the same document in real time.

Microsoft Excel

Excel is a commonly used spreadsheet tool with a wide variety of uses.

LaTeX

Latex is a document typesetting and formatting tool and is widely used in academia. In our project it was used for this report.

Adobe Illustrator

Another tool used to visualize our project, for example mind-mapping, concept design, flowcharts, in addition to logo development for our group.

Gantt

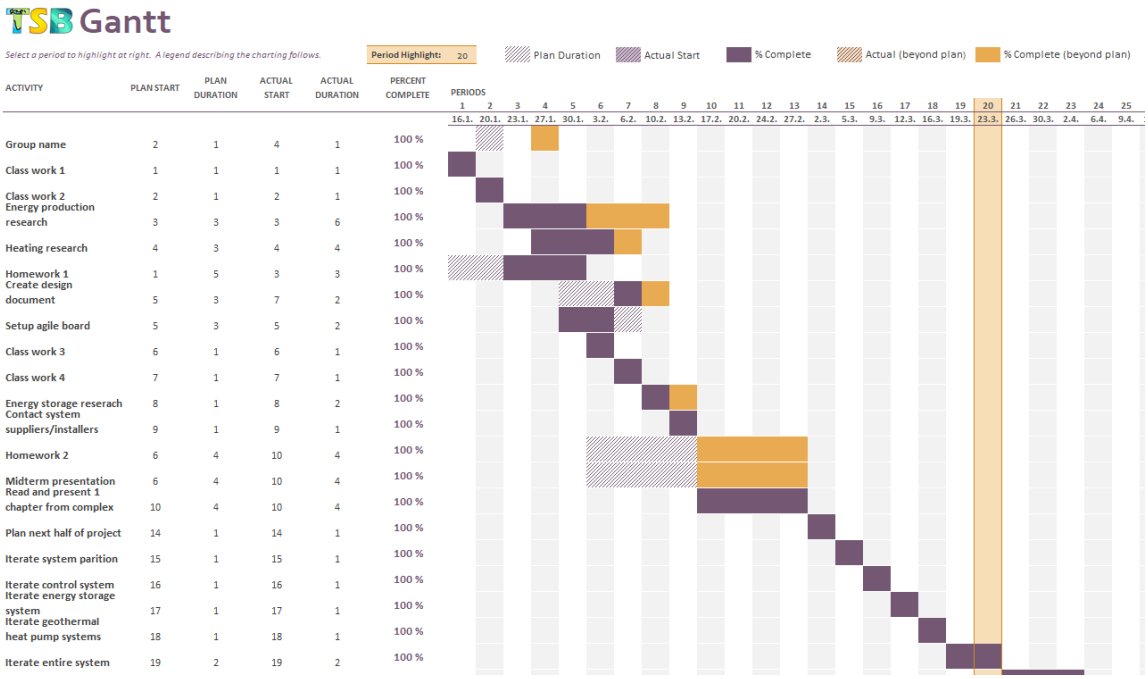


Figure 4: Gantt diagram screenshot

A Gantt chart, commonly used in project management, is a way to view activities displayed over time. On the left side of the chart is a list of activities and along the top is a suitable time scale. Each activity is represented by a bar; the length and position of the bar reflects the start date, duration and end date of the activity. This allows you to see, what the various activities are, when each activity starts and ends, how physician each activity is scheduled to last, where activities overlap with other activities, and the start and end date for the entire project. To our project, having such a chart gives an overview to easier plan ahead and divide the contents among each other.

Draw.io

Yet another tool used to visualize our project, for example mind-mapping, concept design, and flowcharts. This program is online and in browser, for easy access and completion, again making it easier to iterate consecutively.

Sweet Home 3D

An open source modeling tool. Used to draw a 3D model of the building.

Solidworks

Used to model the apartment complex with the energy.

Programming Languages

One can use programming to develop programs to automate all kinds of time-consuming tasks.

Different languages have different advantages, but mainly it is up to the developer and his/her proficiency in the language to decide if it's the right tool for the job.

We have used programming for our machine learning prototype, the website prototype, parse data, communicate with API's, and download data.

We used a mix of Python, C++, Java and PHP.

Github

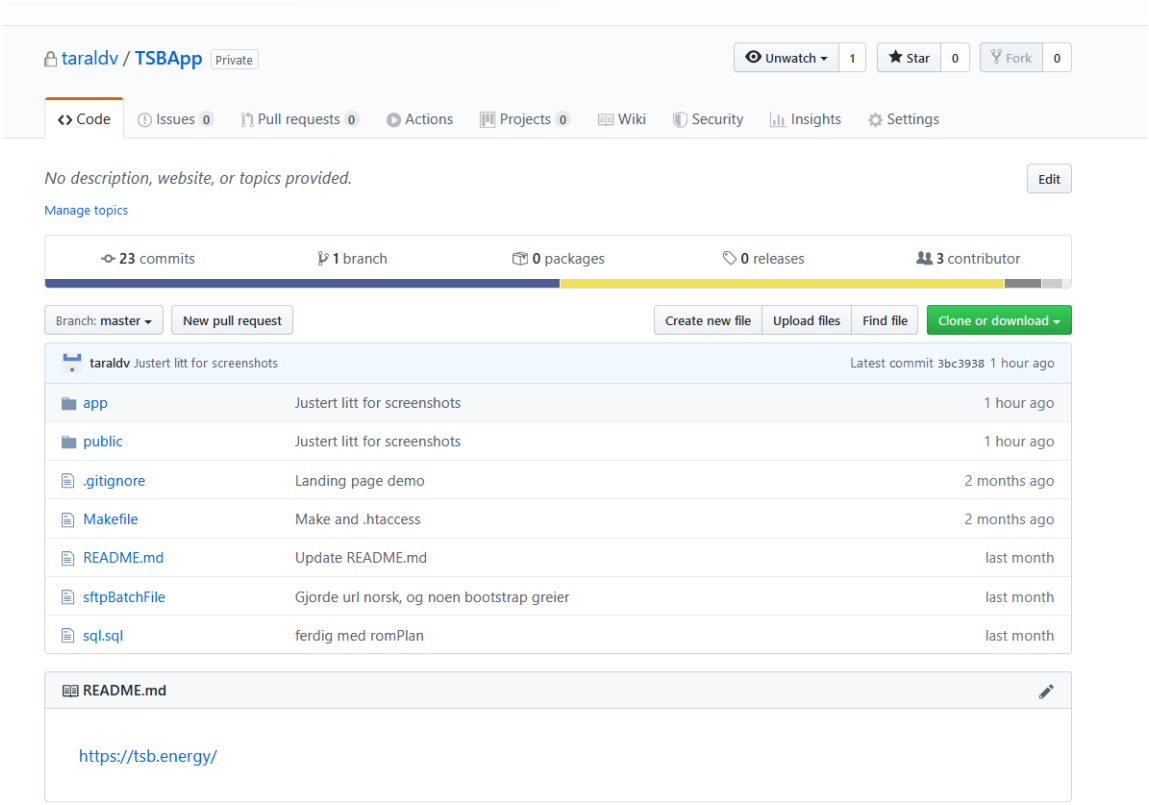


Figure 5: Github web-application repo screenshot

We used GitHub to maintain the codebase for all our programs. It helps multiple developers work simultaneously on the same program.

MySQL

For relational databases, described in detail at page 52

MongoDB

MongoDB is a widely used NoSQL implementation suitable for big data. This helped us prototype our machine learning, and is described in detail on page 52

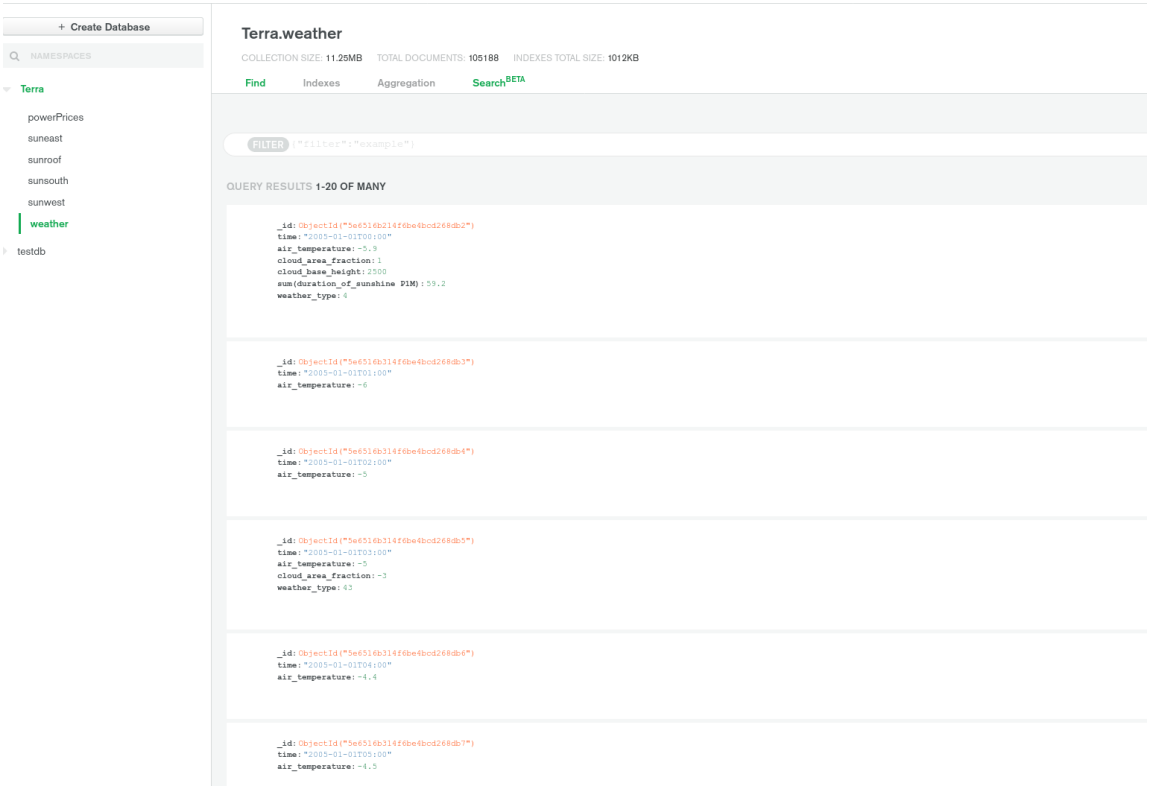


Figure 6: MongoDB screenshot

Internal Discussions

At the very beginning, we started out getting to know each other throughout the first couple of meetings, establishing a system engineer as well, that would have an overview over the whole process. Having an established systems engineer is an advantage so the rest can focus on their areas of interest. In general, a good systems engineer will combine the efforts of all disciplines, while having a leveled head and an overview. They will be focusing on the implementations. They will need an agile mindset. Always being able to move on, shifting focus back and forth to iterate. Never quite nailing down anything, and always keeping an open mind for changes.

As mentioned, we chose to divide the tasks among ourselves to efficiently tackle all disciplines

needed. Being able to have this wide range was not only beneficial in general but enabled us to delve deeper than if we were to handle every subject as a group. As that would include a lot of meddling and internal discussions.

Much like what can be referred to as trade-off, our group had many internal discussions regarding the problem and process itself, leading to new viewpoints and crucial decisions directing us onto a certain path onwards. While yes, we have had to backtrack, as is an essential part of reiterating, but some decisions, such as work ethics, group synergy and division among the group are not relevant to this specific systems engineering project, but more our abilities to work together efficiently as a group/group projects in general.

The Problem

Understanding the Problem

In order to find a solution, we need to properly understand the problem, to do this we made the scope smaller. By placing our building in specific location and making it an apartment complex we made the problem more manageable, now we have set conditions to produce a solution for.

- Location: Oslo
- Number of separate buildings: 4
- Floors per building: 4
- Apartements per floor: 4
- Squaremeter per apartement: 100

Figure 7: The location of our building situated in Oslo

To see the problem from the point of view of the people that are living with the problem and are supposed to interact with the solution, we made this user story:

> *Hans Kammerfast (45), his wife Birgit (46) and their two children lived in a mansion, having three gas cars and oil heating. They lived great, but spent a lot of energy on heating and had difficulties: They struggled with the children, that didn't get to heat their room after the cold night before they had to leave for school. Birgit felt like an environmental pig, and Hans wasn't too convinced whether what they did necessarily was bad for the environment. In addition, Hans was annoyed owning to the unreasonable electricity prices, and it affected the well-being of the whole family.*
>
> *Following the first time Hans really got to see the statistics over his own emissions, he got a revelation. He felt dirty and wasteful. Unhappy with today's environmental decline due to fossil fuels, the family found that they wanted to contribute to a better everyday life. So they are currently searching for something better: a solution...*

When we see the problem this way it illuminates the possibilities to make life easier for the users at the same time as we offer a solution to the problem.

So, who will our solution affect when we take location into account? How do we interact with the municipality? How will our system affect neighbors? What sort of laws do we have to uphold? These are some of the questions we need answers to.

Context

We first made a quick sketch of the building. As it is the most important man-made system our system will interact with.

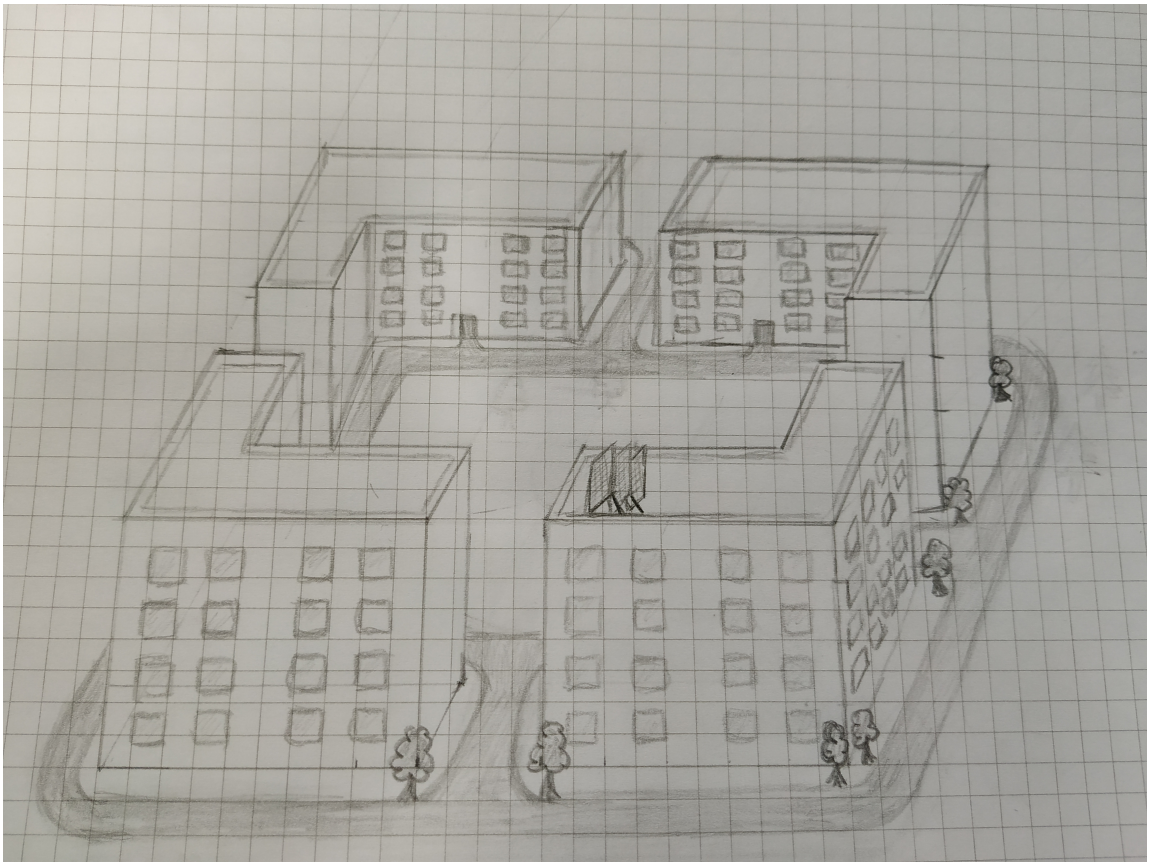


Figure 8: The initial sketch of the context

In order to have any realistic hope of achieving a 100% self-sustainable system, the building will have to be built according to these requirements:

- The building shall fulfill all requirements specified in NS3700
- The building shall fulfill all requirements specified in TEK17

The NS3700 is a standard that specifies requirements for buildings in order to be classified as a passive house. Enova has found that this will only cost developers about 1200 NOK more per square meter[7].

We also made a 3D model of the building to better show our stakeholders how it would affect their building.

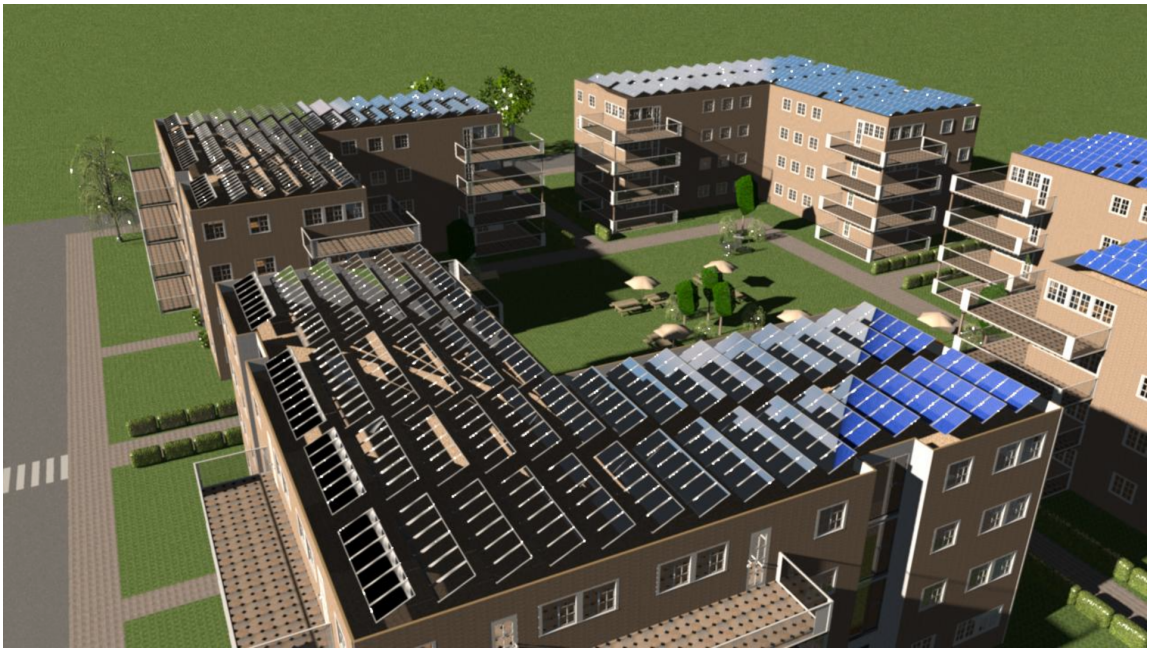


Figure 9: A 3D model of the building

A video of the 3D model can be seen at:
`https://drive.google.com/open?id=1G_30d9ilNBDxjQSbs0yWnLItGcMBMfl5`

We then spent some time finding out all the man-made systems which will interact with our system. We found problems which might ordinarily only be discovered near the end of development or even after the system is finished and placed in the real world.

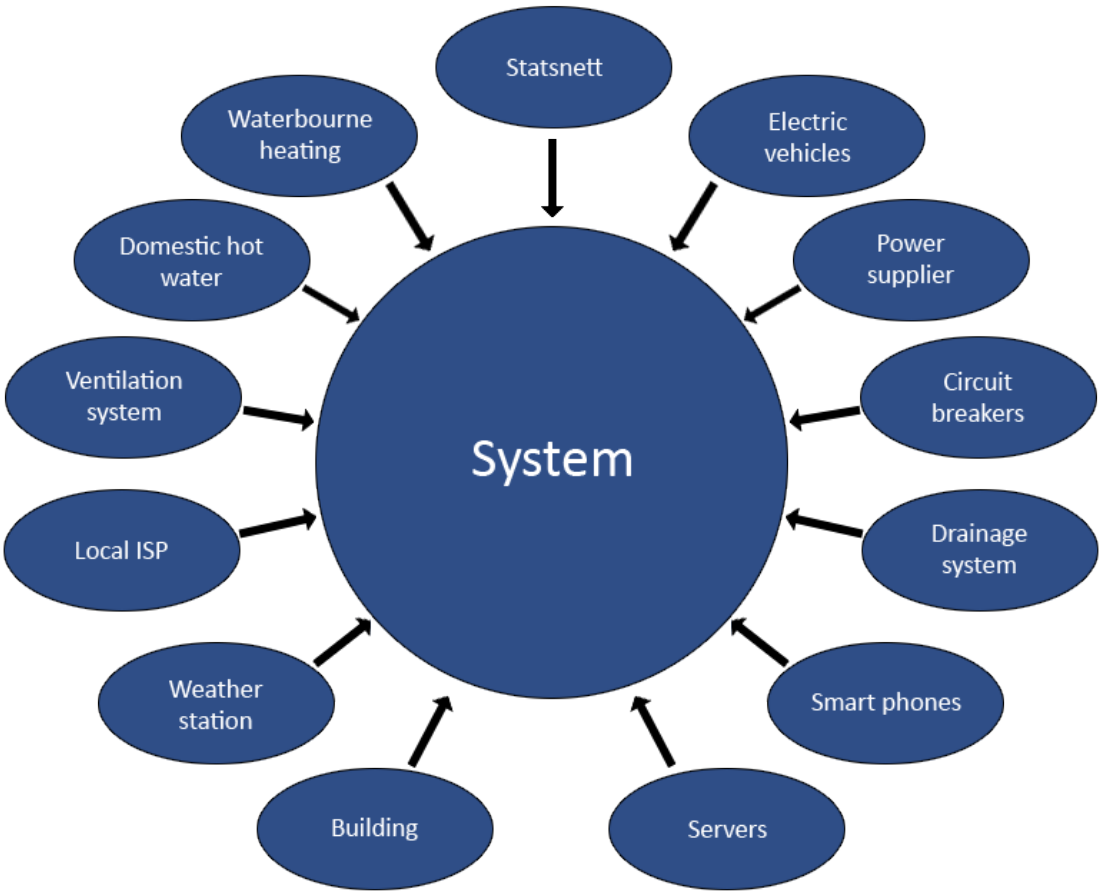


Figure 10: The man-made systems in our systems immediate context

Stakeholders

We wrote down a list of as many stakeholders we could possibly think of.

When looking at our system from their point of view, we find a good solution early. We avoid working on a bad solution because we did not consider, for example, how the neighbor would be affected.

Then we made an onion-diagram to place the stakeholders according to how "close" they are to the system, and how much they are affected by the system.

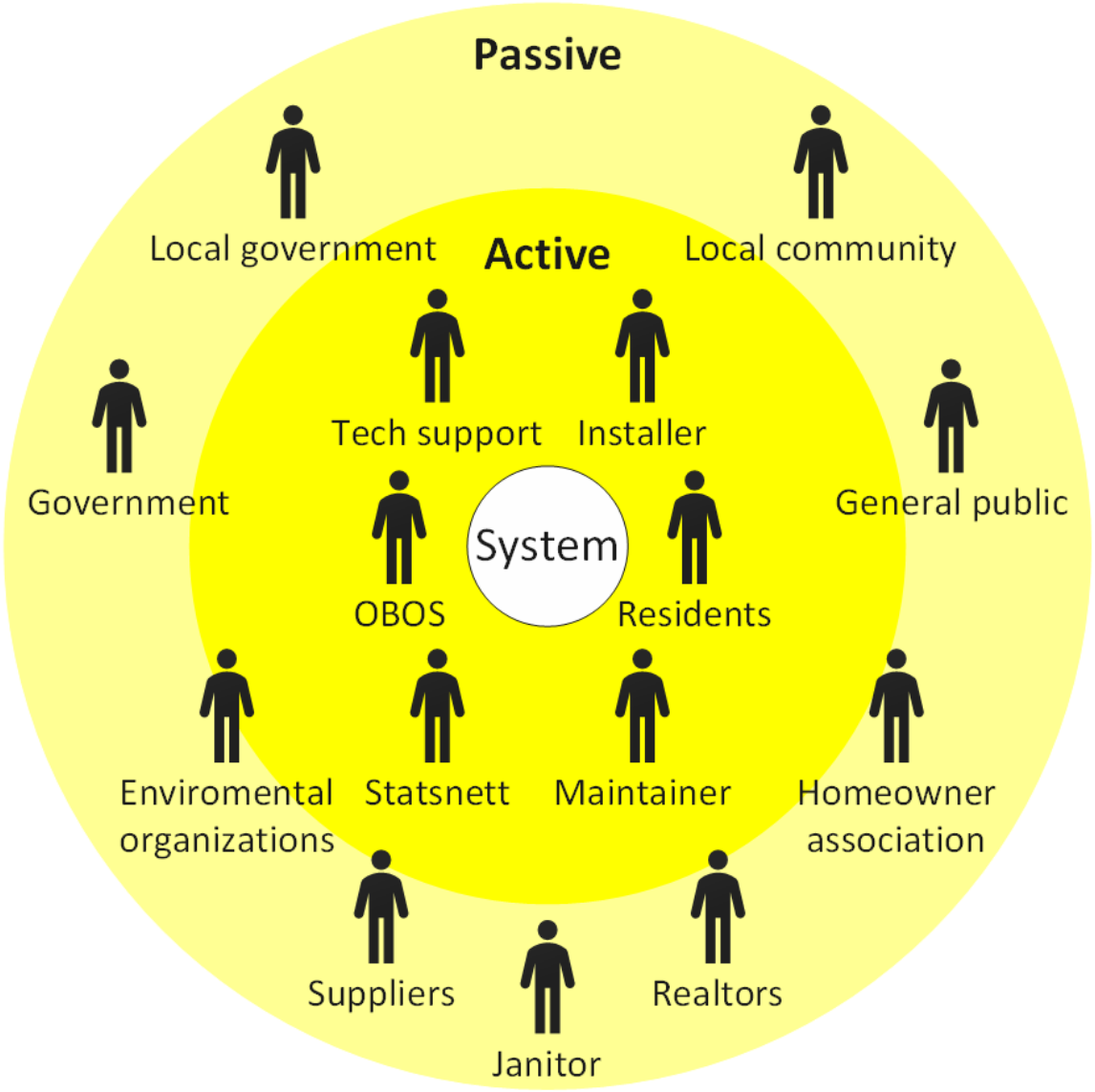


Figure 11: An onion diagram showing the stakeholders

After we determined who our most important stakeholders are and their "proximity" to the system, we started finding their concerns, these are crucial for developing the system, from the concerns of the stakeholders we can develop requirements for our system.

Requirements

User Requirements

This is the Mecca of requirements. The holy grail. Finding these are crucial to understanding what the system should do and be, on a leveled basis. We started out with a viewpoint from a potential resident in our system/apartment complex in addition to scouring the internet for similar viewpoints to gain insight of the situation. With these viewpoints, we could find certain key words and phrases that were important enough for our group to focus on in our system equation. We were also asked to focus on the viewpoint of another type of stakeholder, which we decided was going to be OBOS. (As mentioned earlier, stakeholders) In our "constructed situation", OBOS had hired us to take on this task.

Stakeholder concerns

1. OBOS
    - Marketability
    - Enviromental
    - Profit margin
    - Installation time
    - Reliable
    - Regulations
2. Residents
    - Affordable
    - Enough energy
    - Non-intrusive service
    - Prestige
    - Adjustable temperature
    - Enviromental
    - User friendly app
3. Statsnett
    - Energy demand
    - Energy supply
4. Installer
    - Simplicity
    - Safety

5. Maintainer
    - Easy access
    - Good documentation
    - Safety
6. Tech support
    - Good documentation

As they were, these user requirements were still important to our system, but not practicable to implement into the system itself, which we needed to do.

Analysis of Competition

After finding some main guidelines, we chose to research our competitors, both immediate (other groups with the project) and real (established companies), to compare and assess the reality of the situation. The research gave us clear trends, opportunities and problems, that are both outside of our system, in addition to potential problems our system could have.

Some reoccurring trends were the focus of "Green energy", having a clean conscience and future planning, where future planning is including the distant future's impact on the product in the calculations. We understood this was a necessary part in the development of our project, so we used that to make both a schedule for the lifetime of our product in addition to finding some new requirements.

As with trends, there were some opportunities for our group, with companies wanting new blood and fresh eyes to keep up in an ever-changing society.

The problems with the market in which our group are resigning in, is that it is saturated with large multimillion-dollar companies, who are trusted in what seems to be something among the lines of monopoly, with an example being the Ramboll group (engineering, architecture and consultancy company), involved in many Norwegian projects like Vestsiden ungdomsskole in Kongsberg.

In addition to these, we also found some internal policies for an example company connected to engineering as they were using an approach called RAMS. RAMS is a characteristic of a system's long term operation and is achieved by the application of established engineering concepts, methods, tools and techniques throughout the lifecycle of the system.

European Railway Standard: "In railway engineering, as in most forms of transport, safety is a paramount. However, it is influenced by, and interlinked to, availability of the system. Availability is influence by operations and maintenance practices as these determine the reliability and maintainability of the system."

Effectiveness Measures

Our group had a proper discussion regarding what the difference of user requirements and effectiveness measures were, differing between chosen user requirements that by themselves, as they are "make-or-break" the system, and a reformulation of the user requirements more suitable to use for finding the next level of requirements more practicable for the system. No matter the difference, we made the decision to trust in what teacher Gerrit implied, from which we understood it, and find the most important user requirements that "make-or-break" the system so they could be our effectiveness measures.

INSERT EFFECTIVENESS MEASURES

We used these effectiveness measures to find the more practicable requirements for our system. We used a method called key driver graph.

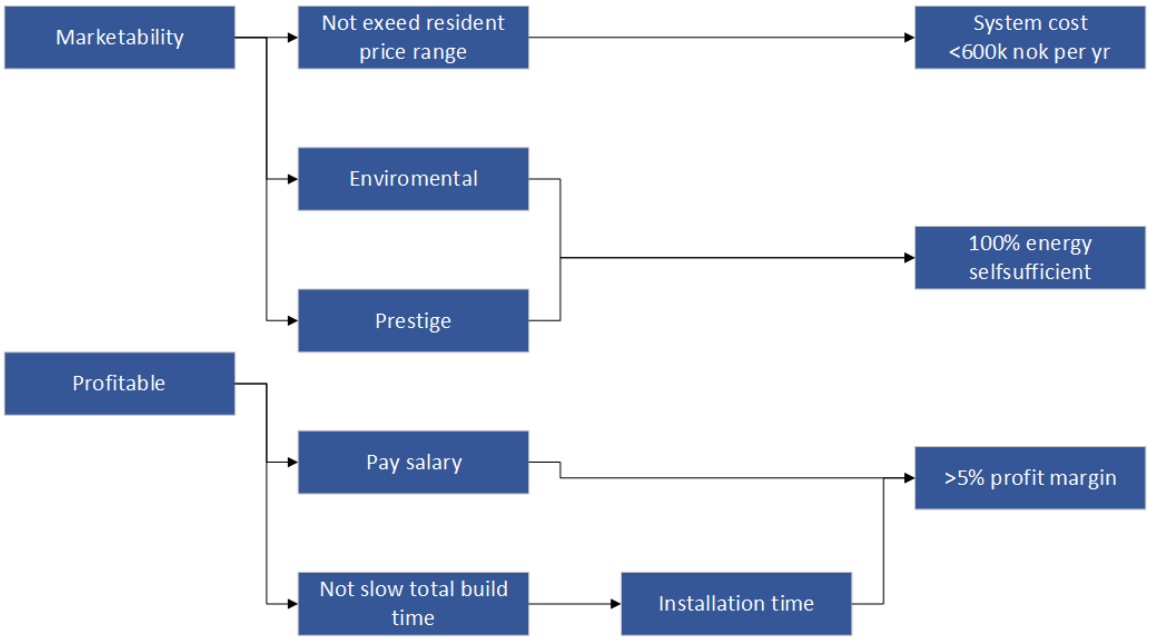


Figure 12: Key-driver-graph for OBOS

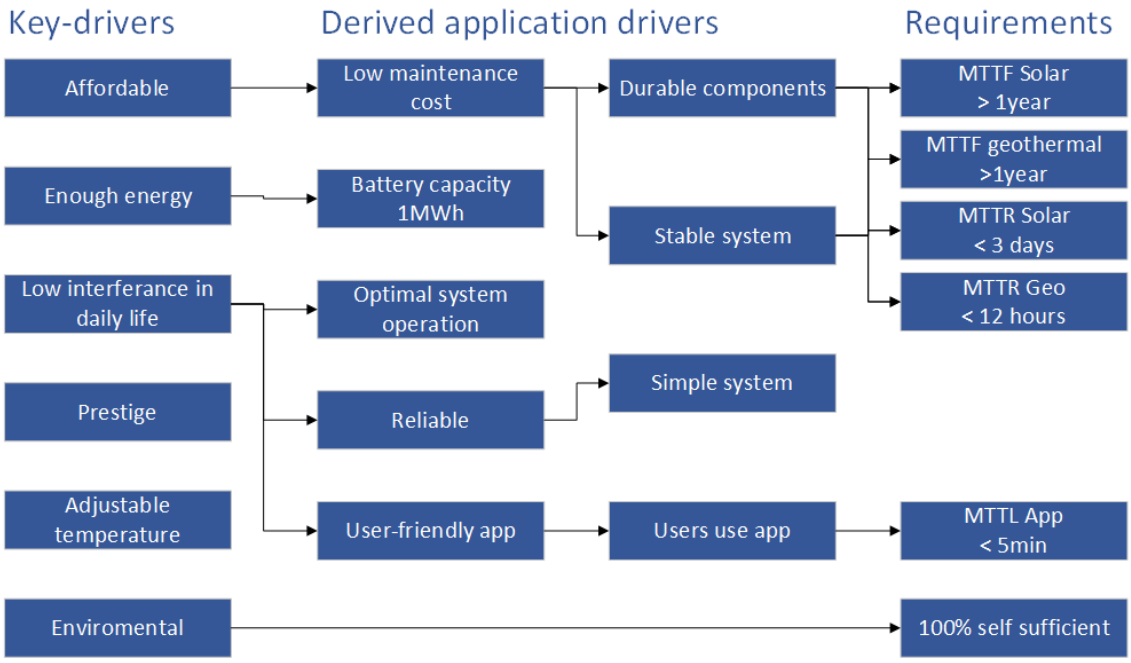


Figure 13: Key-driver-graph for the users of our system

OPPDATERE KEY DRIVER GRAPHS In this diagram the main goal is to find quantifiable numbers to use in our system equations which we can cross check and implement later.

From this diagram we found for example the mean time to failure (MTTF) of our system should be XXX, which changes the system in a mechanical way, perhaps changing the price range, in addition to ensuring a longer lifetime of said part. It is important to pay attention to the make-or-break requirements in this way as they yield some of the highest gains one could get.

System Requirements

DENNE DELEN BØR SEES OVER

Now, one can divide these system requirements into several sub-categories;

- Functional
- Temporal Performance
- Non-Temporal Performance Requirements

- Interface Requirements
- Design Requirements

Our focus was to find Key performance parameters (KPPs), as they were referred to as. In reality, these parameters are a mix of the other types of requirements molded and mixed together.

REF REQUIREMENTS TIER 0

Energy Consumption

The energy consumption of the apartments and the whole building complex will decide what kind of system is viable.

Finding numbers is important when it comes to trade-off analysis. But as students we are not always equipped to simulate all our estimates. We did not want to spend too much time learning different software to simulate a buildings energy consumption. We wanted to focus on the subject at hand, which is system engineering.

For our estimates we decided to use the Norwegian building standard Tek17 as a guideline. It contains energy consumption regulatory limits for new buildings. It has maximum numbers per m2 for ventilation and total energy consumption [17][10].

For heating we used another Norwegian standard which has maximum numbers for heating per m2[11]

For lights and appliances, we used numbers from an average home gathered by a popular power supplier in Norway, Hafslund strøm. The average light in Norway is not LED however. So, we divided the light consumption by 10. [16]

Electrical vehicle charging consumption is estimated from how far an average personal vehicle in Norway drives per year. Which is 12000km [9]. And how much energy an electric vehicle use per 10km driven. Which is between 1 and 2,5kWh[15]. We decided to estimate on the safe side and use 2kWh per 10km driven.

For elevators we used a popular calculator online and used the most energy efficient solution.[6]

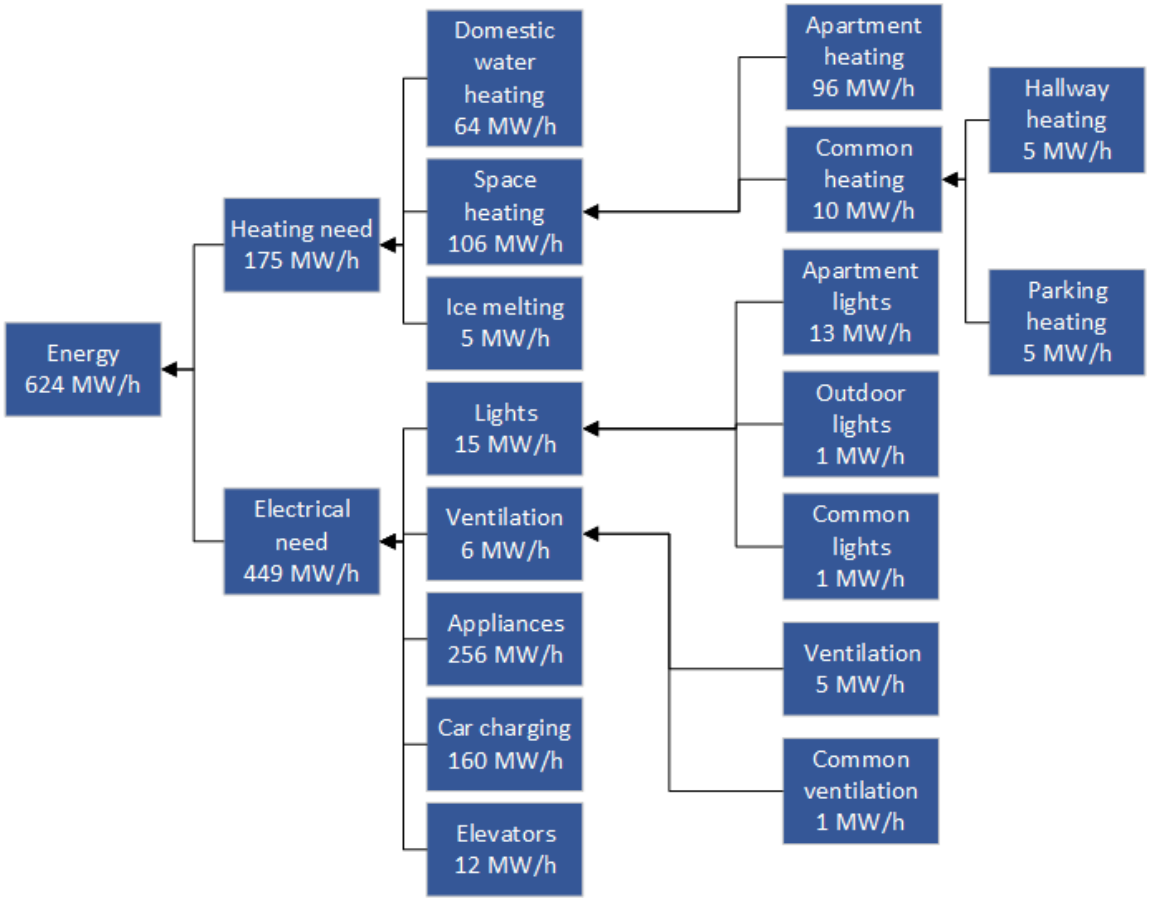


Figure 14: A technical budget displaying the energy needs of the building

System Architecting

We will now look at the architecture of the system. This is a process which takes place on many levels or tiers. At tier 0, we will look at the system in the highest level of abstraction. This means that one views the system in its context and avoid considering details within the system.

Later, in tier 1, we will look at the subsystems, how they relate to each other and the dynamics between them.

Finally, in tier 2 we look at the subsystems and components within the subsystems which we have considered critical, and therefore focused especially on.

Tier 0: The System in Its Context

Operation

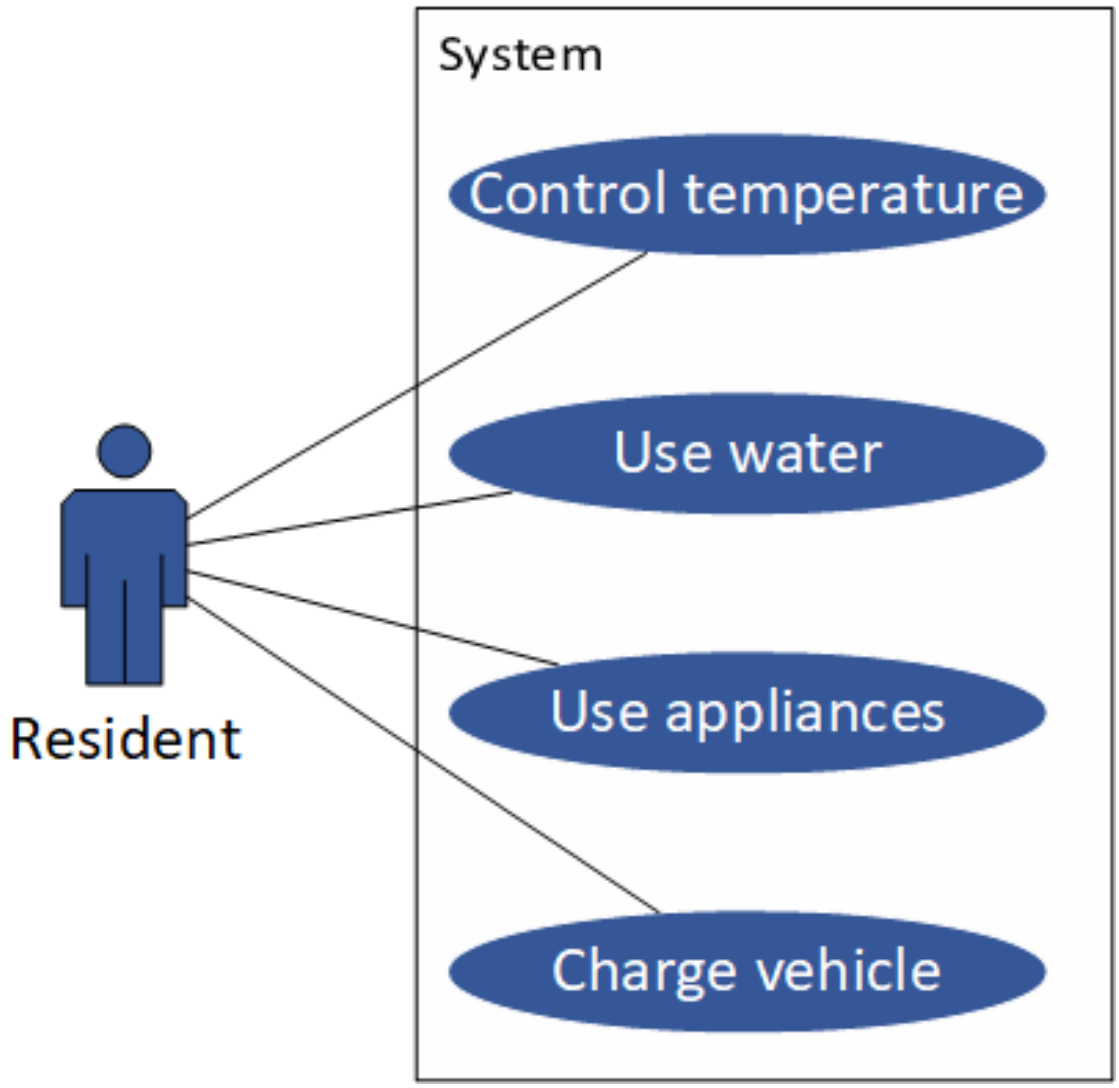


Figure 15: Use case diagram for the system

Here we can see at the top level what the base use cases of our system will be for its users. They will derive the systems inputs and outputs, as well as the functions we need in our system. Doing it this way, helps us make sure that the system fills the needs of our user, as well as avoiding adding

functionality that doesn't fill a purpose.

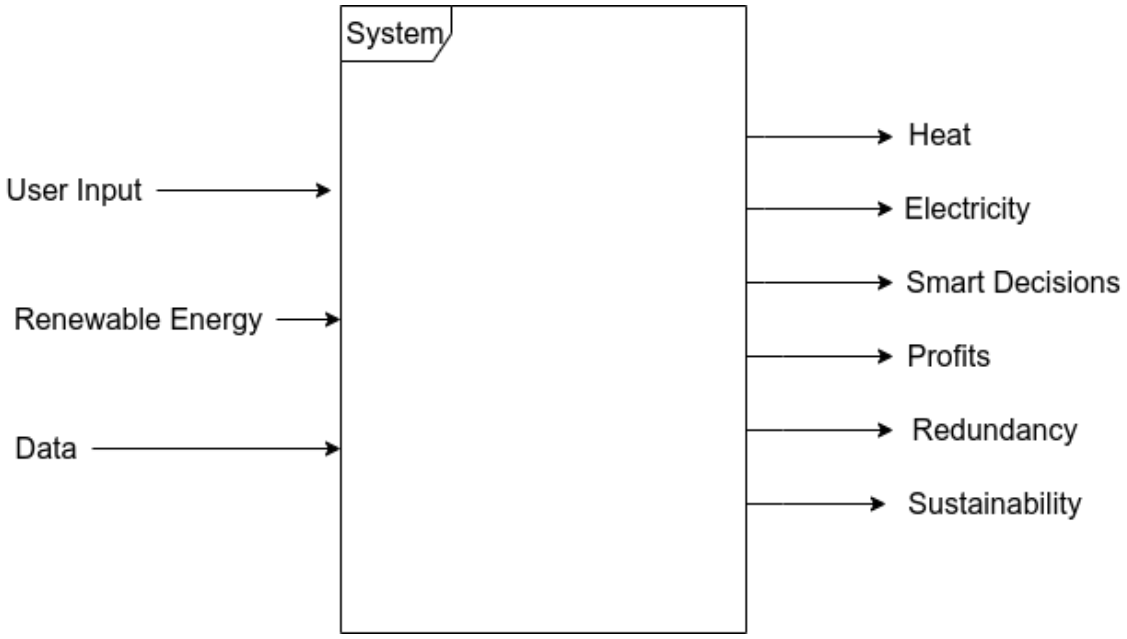


Figure 16: Black box of our system

In figure 16 we see the system as a black box. We found that the system will receive/gather renewable energy, needs from the user as well data from various sources. Then we identified the wanted outputs. These were all derived from our investigations into the customer, the users as well as the context of the system. This helped us explore what functions our system will need to have.

Behavior

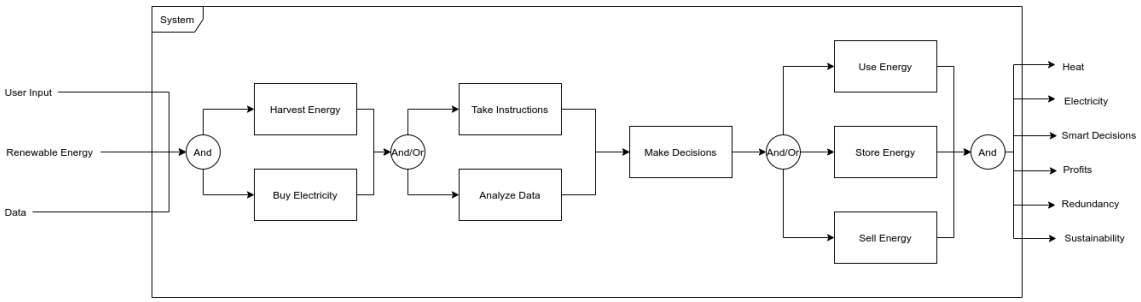


Figure 17: FFBD within our system

In order to provide heat and electricity in a sustainable way, we will need to harvest energy, but we will also need the option to buy power from the grid. To make the system smart, it will need to make decisions based on both data and the wishes of the user. Finally, it will need to allocate the energy in an optimal way in any given moment, that may involve using it, storing it, or even selling it.

Structure

This structure of the system describes how it will be built. At this point we start with dividing it up into subsystems:

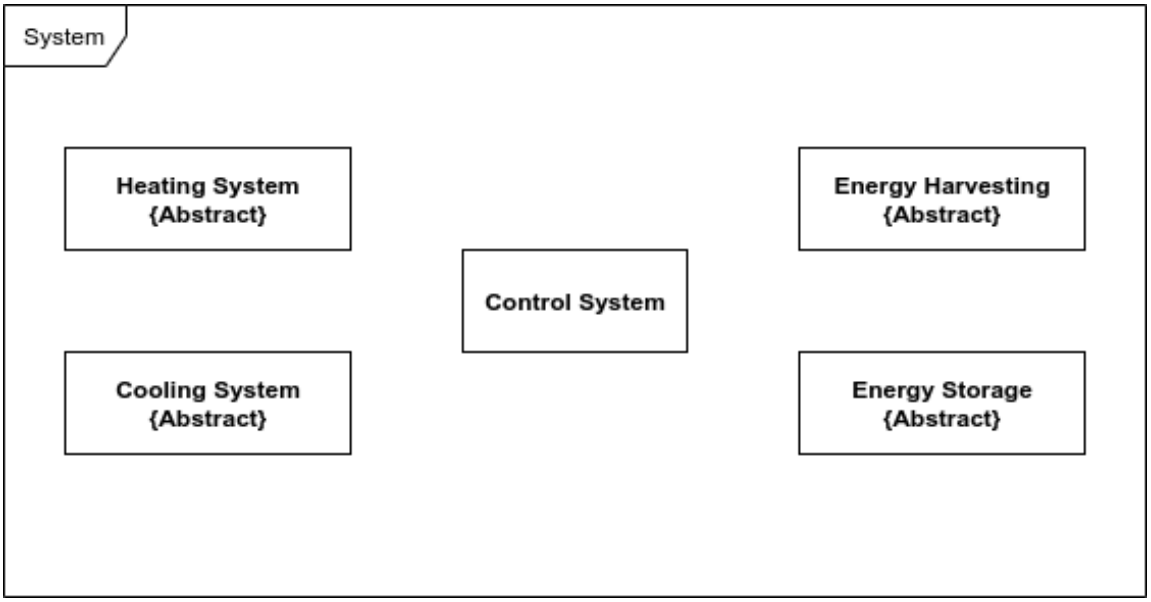


Figure 18: The system modeled as a UML class diagram, where all the classes are still abstract

There are many technologies available for us in order to make our system attain the wanted functions and behavior. The different combinations of them gives us thousands of possible concepts. These are the options we initially considered, where we have looked closer into some of them:

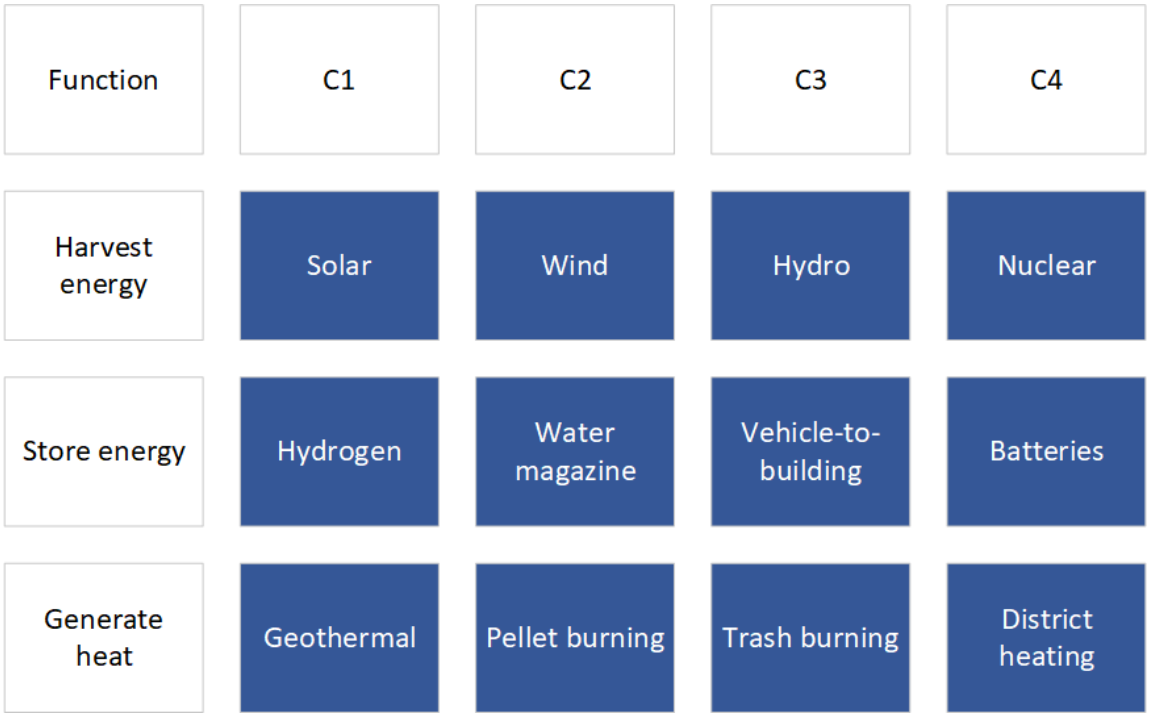


Figure 19: Morphological diagram, showing possible concepts

Harvesting Energy

Solar

Solar-power is growing at a tremendous rate. It is easy to implement and is getting cheaper by the day. Given the context of our system, one may think that the Nordic climate wouldn't be favorable to this technology. However, even though we have less sun here, the cold makes the panels more economically efficient in Norway, than in for example a place like Milano[8].

Hydro

The earliest recorded use of hydro for harvesting energy is from the times of the Roman Empire, and is very widespread today, especially in Norway. It is a very mature technology, can produce massive amounts of energy and the potential worldwide still is not near begin filled. However, in Norway the government owns the right to harvest energy from almost all viable locations, which makes it hard to implement for a private residence. The locations of our customer group also makes it difficult as the availability of waterfalls and rivers is even less in urban areas.

Wind

Harvesting wind-energy is another very old approach to relieving humans of manual labor and has in modern times become a means to generate large amounts of electricity. A single windmill with a diameter of 120 meters, can generate up to 4.5 MW on average [4]. Because of this they been become very widespread the last 20 years, even in Norway. However, because of their size, the noise they generate and the dangers they present to the fauna, windmills have become the source of huge political controversy lately.

Our main customer group are in urban environments, and this means that windmills will be difficult to implement in an effective way. There does exist so-called urban windmills to be placed on rooftops, but the electricity they generate is most often not worth the cost.[3].

Nuclear

Nuclear energy has a bad name, even though it produces completely clean energy. The trouble with nuclear waste storage, as well as public fear of accidents makes this option a practical impossibility. It is extremely expensive to realize, not renewable, and requires a much larger scale, than we can facilitate.

Storing Energy

Batteries

For storing the electrical energy our system will generate, the most obvious option is lithium-ion-batteries. They are popular because of their high charge density, as well as their effectiveness. They are simple to implement but have a high cost.

Another problem with them is that they are made from non-renewable elements, and the world is running out of cheap and accessible lithium. The process of manufacturing them is also, not considered to be environment friendly.

Vehicle-to-Building (V2B)

Vehicle-to-building is a concept where you connect a two-way charger to an electrical car. It lets us use the car-batteries to both store and fetch electricity when needed. This can reduce the needed capacity of other energy storage and may help reduce the costs of the system. In order to make use of it of course, you will need electric cars, but this does fit our customer-group very well.

Hydrogen

Hydrogen can be used as an energy carrier for many different applications, often releasing the energy stored in the hydrogen gas in a fuel cell. Hydrogen can also be burned in an engine, but the industry has largely departed from that technology, since fuel cells convert energy much more efficiently.

The benefits of hydrogen as an energy carrier are that hydrogen per mass has an energy density that is three times as high as traditional fuels such as gasoline and diesel, and that it does not result in emissions other than water vapor when hydrogen and oxygen are converted into electricity in a

fuel cell. The challenge is that hydrogen is a very voluminous gas and is therefore more demanding to handle and distribute than traditional fuel. There is an opportunity to solve this challenge by producing the hydrogen gas in the same place it is consumed.

Water Magazine

This approach basically consists of pumping water to a magazine at a higher altitude. When you want to 'withdraw' the energy, you release it, and funnel it through a hydro power generator. For obvious reasons this will not be a viable approach for us, because of our scale and location.

Generating Heat

Burning Trash

Burning trash may initially sound like a bad idea, but it is a quite widespread way of generating heat and electricity. Most notable in Viken is perhaps the plant in Fjell, Drammen. Fjell is a suburb of Drammen and consists of a large number of apartment complexes. Their heat facility provides all of them with heating exclusively from burning trash. The waste of this process is pure ash, and this is sold to businesses who use it to produce concrete and asphalt. The method is generally seen as environmental-friendly, as it mostly releases steam into the air. However, it does release several other elements into the environment like NOx, SO2, CO and mercury, though in small amounts. This may make it unsuitable for our purposes.

Another problem with this approach is that, in order to be economically viable, you will need quite a large-scale facility. In other words, it does not fit the problem we are trying to solve.

Geothermal

Geothermal energy is often separated into two categories. The first one is the concept of drilling down to the boundaries between tectonic plates and extracting the extremely high temperatures found there. If this is ever accomplished in practice, some prophesize that is will generate enough energy for the entire earth, several times over.

The concept we are looking into is ground source heating. It involves drilling holes in the ground, and putting tubes filled with brine in them. They can be anywhere from six meters, to several hundred meters deep, dependent of the area. You also need a heat pump, which will pump cold brine down, and extract the heat located in the ground. The brine that reappears in the other end, will have a constant temperature, year-round. The heat pump will then use the gain in temperature to heat water in a much more efficient way then it would otherwise be able to. The process can also be done in reverse, and in that way provide cooling in the summer. State-of-the-art heat pumps can even provide heat with an effectiveness factor of 5. This means that for every kwh of electricity you put in, you may get 5kwh of heat out. This has been a popular concept for existing apartment buildings lately, because Norwegian law banned all forms of oil and paraffin heating in all buildings from 01.01.20.

Burning Pellets

This option is basically the same as burning trash, but instead of trash we burn pellets made of biomatter that are compressed. In contrast to burning trash, this option is much more viable in a small scale, and there are companies specializing in delivering pellets to apartment complexes today. However, we found that in urban environments they release particles into the air, which makes for a worse environment for people in its surroundings. This will probably not meet the profile of our system, nor will it make it easier to market.

District Heating

Probably the most widespread way of heating apartment complexes in Norway today, is district heating. It provides cheap, reliable heating from external, most often local plants. They vary in their environmental factor, but most often they are considered to be environmentally friendly.

Trade-Off

In this section we will have a look at the options and analyze which ones may be best suitable for our system.

Harvesting Energy

We looked at a few different ways of harvesting energy for our building complex. Considering that one of our effectiveness measures is to provide green energy, we could narrow our options down to Solar, Thermal-Generator, Nuclear, Wind and Hydro. When it came to finding out which one of these options would be most suitable for our system we had to look back at our measures of effectiveness and requirements. We picked out a handful of these that were most relevant to energy harvesting and set up this PUGH matrix:

Criteria	Solar	Nuclear	Wind	Hydro	Thermal-Generator
Sustainability	5	3	5	3	2
Cost	5	1	4	3	2
Difficulty	5	1	5	3	3
System complexity	5	1	5	3	3
Lifetime	3	5	3	3	3
Improvability	1	2	1	2	2
Support	4	1	2	4	3
Outsourcing	4	2	4	4	3
Controversy	5	1	3	5	3
Availability	5	1	5	1	3
Safety	5	5	2	5	2
Efficiency	3	5	3	3	4
SUM	**50**	**28**	**42**	**39**	**33**

Figure 20: Pugh matrix for harvesting energy

By using this PUGH matrix, we were able to evaluate our options and we concluded that solar panels would be best suited for our system. It would not only be best suited for this specific system with the current location of medium density Oslo, but it's also the option that would be easiest to implement at any other location. Comparing it to the nuclear option, it's clear that solar is way more doable in terms of cost and difficulty. And if we compare it to wind and hydro, solar panels do not depend as much on the location and outside factors as these other options. By choosing solar panels we make it easier for us to distribute our system virtually anywhere.

For a large building complex with multiple floors the current generation solar panels do not provide enough energy to provide 100% of the building consumption. But with four floors it is close enough. In Norway renewable energy from the grid is very easy to guarantee. We will simply change our goal from 100% to 60% self-sustainable. And get the other 40% from a power supplier with a guarantee that is comes from a renewable source.

Storing Energy

Since our system is harvesting energy we need some way of storing it. There are several forms of having grid energy storage. Among them are batteries, compressed air, hydrogen and electric vehicles. Some of the important factors we had when choosing a storage method was: sustainability, we want to leave as little of a footprint as possible, the cost of it, we want it to be affordable for our customers, how safe it is and how efficient it is, we want to minimize the energy waste. From researching different methods of grid energy storage we picked three options we thought would fit our system most and compared them using this PUGH matrix:

Criteria	V2B	Hydrogen	Only batteries
Sustainability	2	5	3
Cost	4	1	3
Difficulty	2	1	5
System complexity	2	3	4
Lifetime	2	4	2
Tenant capacity	2	3	3
Location	3	5	5
Independency	3	2	4
Improvability	4	5	3
Support	1	2	5
Outsourcing	5	1	1
Parking	5	1	1
Park	4	5	5
Availability	4	3	5
Strength	5	3	4
Safety	5	2	5
Uniqueness	5	4	1
Solar cell area	5	2	2
Energy need	2	3	3
Efficiency	5	2	5
SUM	**70**	**57**	**69**

Figure 21: Pugh matrix for storing energy

As one can see from the PUGH Matrix, Vehicle-to-building (V2B) won with a slight edge over batteries. We decided to go for V2B, but because this option has some irregularities in that we cannot guarantee that the amount of cars we need for storage is available at any time we opted for a solution that incorporates lithium batteries as a backup.

Generating Heat

If we want our system to meet our effectiveness measure of minimizing energy waste, we will need a way of generating heat without the use of electricity. We looked into using Geothermal heat pumps, Waste-to-energy and pellet burning. Then we used our requirements to set up a PUGH matrix with the relevant factors we needed to evaluate the options.

Critera	Waste-to-energy	Geothermal heat pump	Pellet burning	Electricity
Heat generation / price	4	5	3	1
Size	2	3	4	5
CO2 emissions	2	4	3	5
Electricity/heat ratio	4	5	4	1
Renewable	3	5	3	4
Cooling synergy	1	5	1	1
Safety	1	4	1	4
Lifetime	3	3	3	5
SUM	**20**	**34**	**22**	**26**

Figure 22: Pugh matrix for generating heat

When comparing our options, we found the geothermal heat pump to be the most optimal solution. It is the only solution among our chosen options that has any synergy with our planned cooling feature, which is an important aspect to the system as a whole. There is also the safety concern in which Waste-to-energy and Pellet burning does not come well out of, since both options poses a fire hazard for our system. Space is a major concern for our system, and we do not want to spend unnecessary money on increasing our heat generation facilities. Pellet burning proved to be the subsystem that required the least amount of space, but the geothermal heat pumps other redeeming qualities weighed up for the larger size.

Recommendations

After analyzing our options for the power harvesting, power storing and heat generating subsystems, we decided on a solution that we would consider to be the most viable for our system.

The solution would consist of solar panels for our power harvesting. A combination of Vehicle to Building?(V2B) and lithium batteries for our power storage. Geothermal heat pump for our heat

generation.

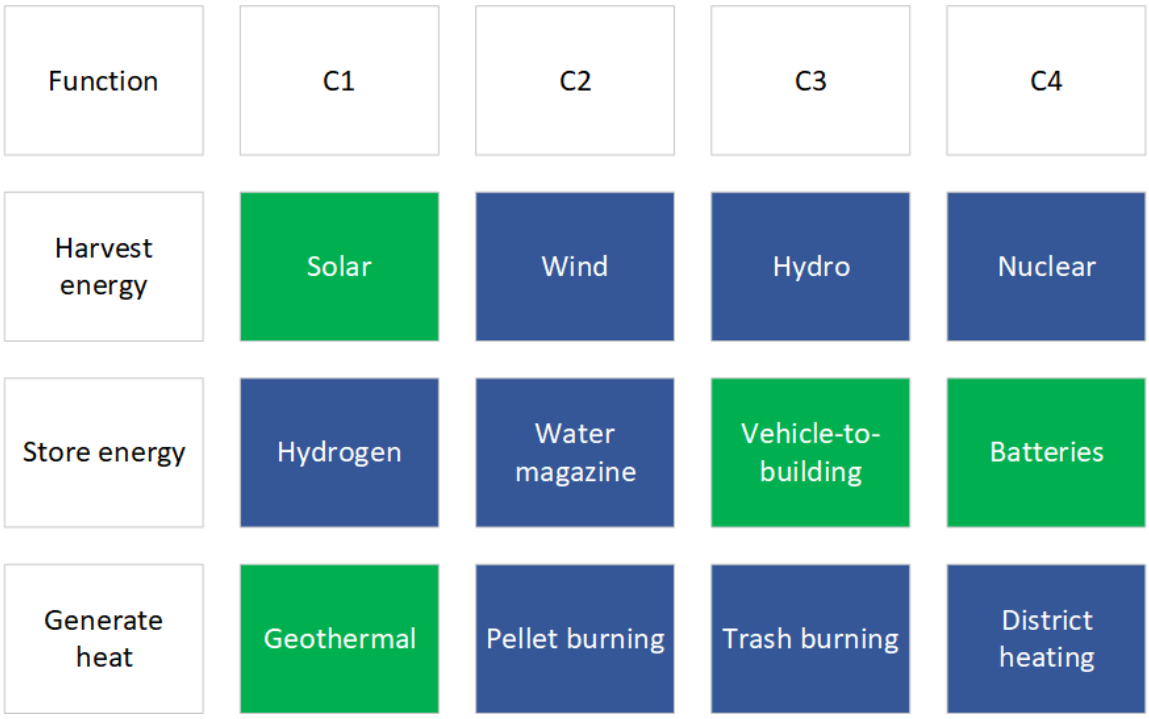


Figure 23: Morphological diagram, showing the recommended concept

Derived Requirements

REF tier 1 requirements

Tier 1: The System

At this tier we are focusing our look inwards try to look at the dynamics and structure between the subsystems in our system.

Operation

These are the use cases found at the system level:

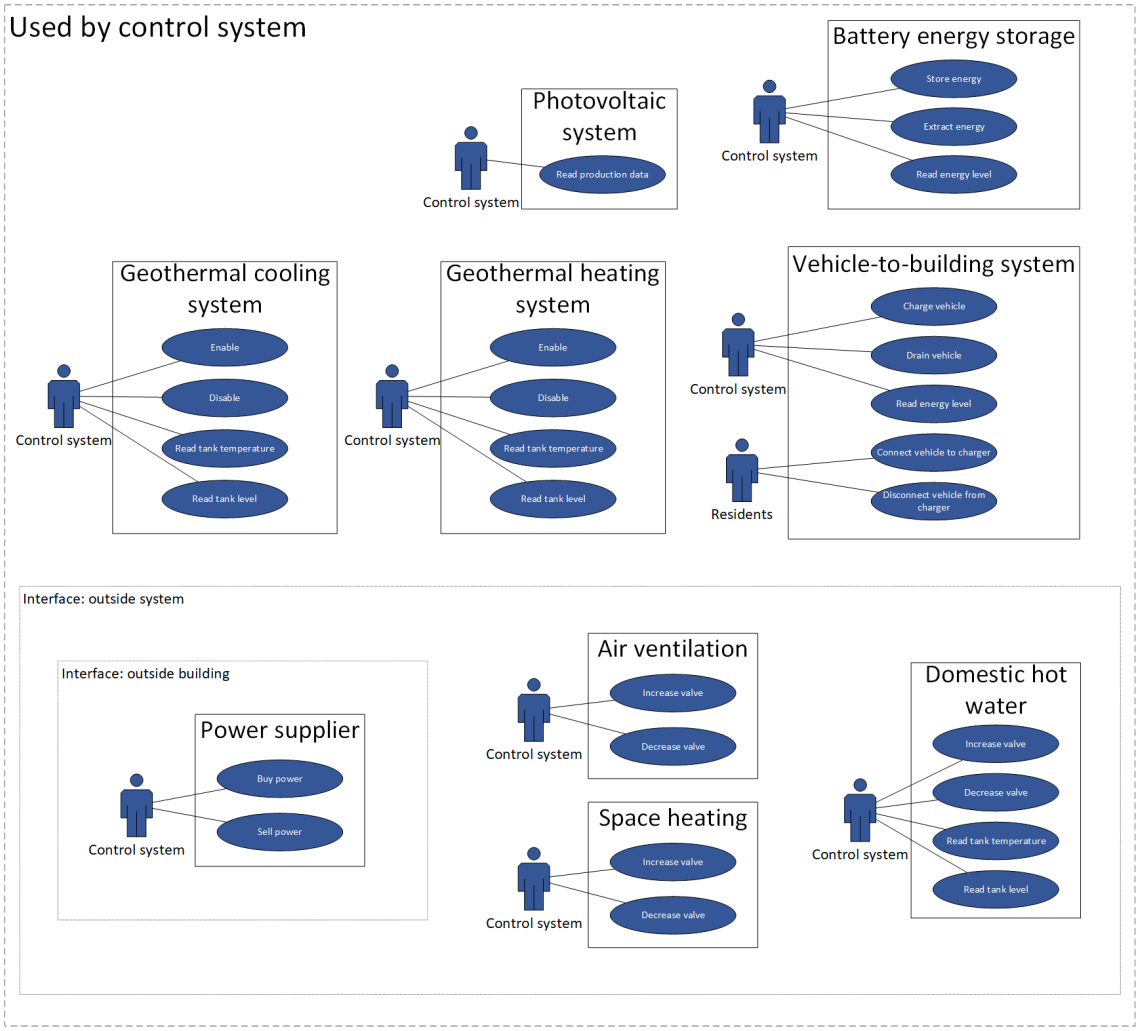


Figure 24: Use case diagram, between subsystems

These use cases were crucial to define the wanted behavior of the subsystems, and how they should relate to each other.

The design choices made at tier 0 will at this point function as requirements. This is a natural phenomenon in the unfolding of a system. Analyzing the behavior of the system is most easily done by looking at the abstract functions of its subsystems.

Control System

A control system is essential in order to fulfill the requirements of the system. This is something we will have to develop ourselves, and is what will make our system smart, as well as giving us an edge on our competitors.

The black box exercise left us with these inputs and outputs:

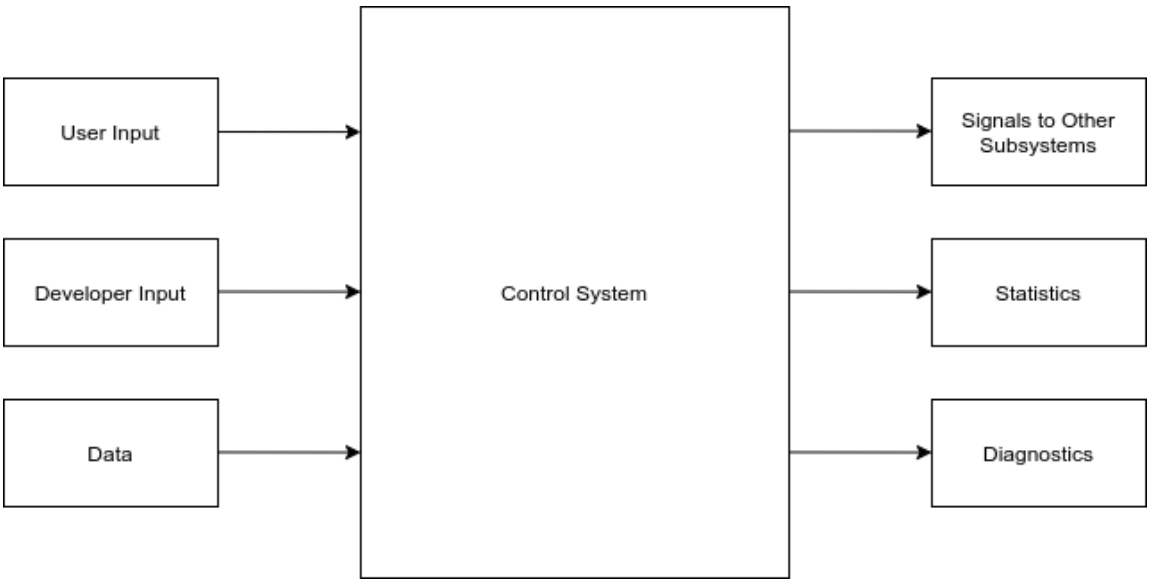


Figure 25: Black box of Control System

As the control system is the main focus of our engineering efforts, this will be further elaborated in the next tier.

Photovoltaic System

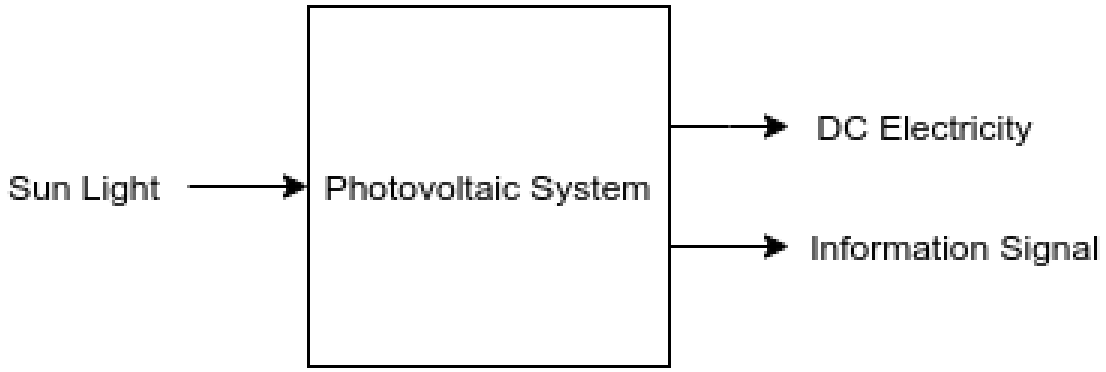


Figure 26: Black box of Photovoltaic System

This is the whole functionality we need from the photovoltaic system and will be bought as a complete package from our supplier. The only thing that needs designing is the software to read the signals, which we will get back to later.

Energy Storage system

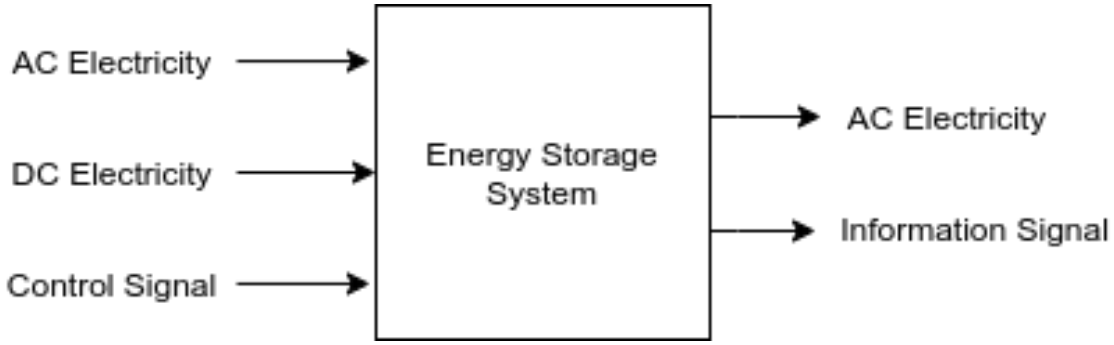


Figure 27: Black box of Energy Storage System

As our concept takes several storage approaches in use, this will be elaborated in the next tier as the allocation of the energy harvested will be crucial for success.

Geothermal Heating/Cooling

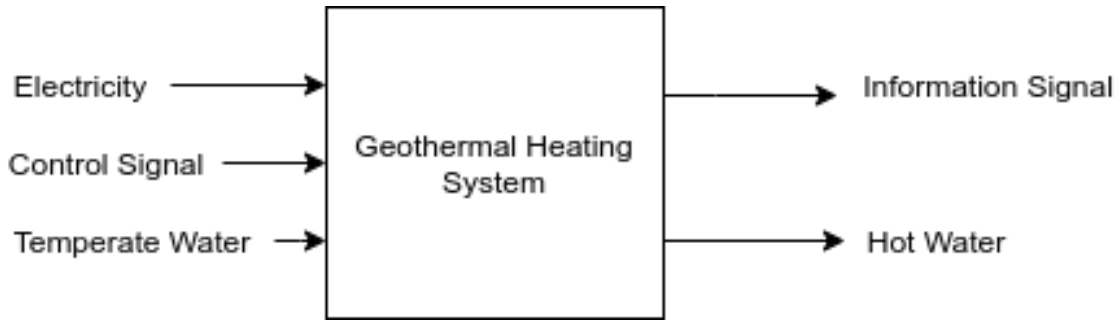


Figure 28: Black box of Geothermal Heating System

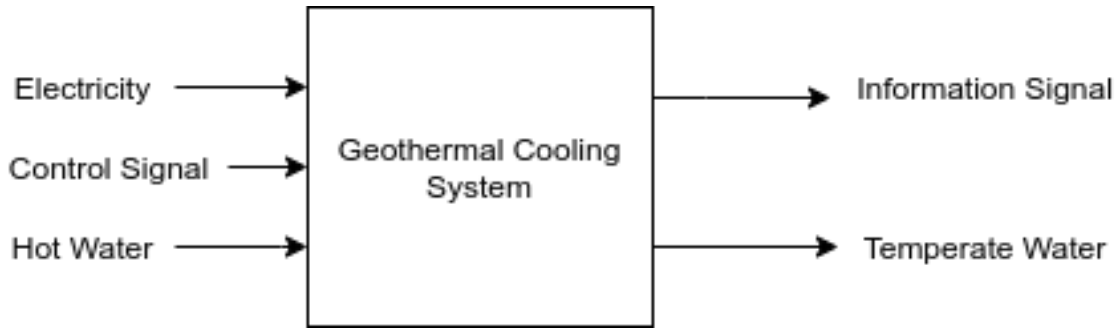


Figure 29: Black box of Geothermal Cooling System

The geothermal heating and cooling systems will receive a signal which tells the heat pumps to either turn on or off. The signal will come from the control system and will be based on different sensor readings throughout the buildings.

Behavior

In figure 30 we can the the functions within the subsystems, that are necessary to achieve the inputs and outputs we need from them. We see how they will relate to each other, and how the outputs of a subsystem, may be the input of another.

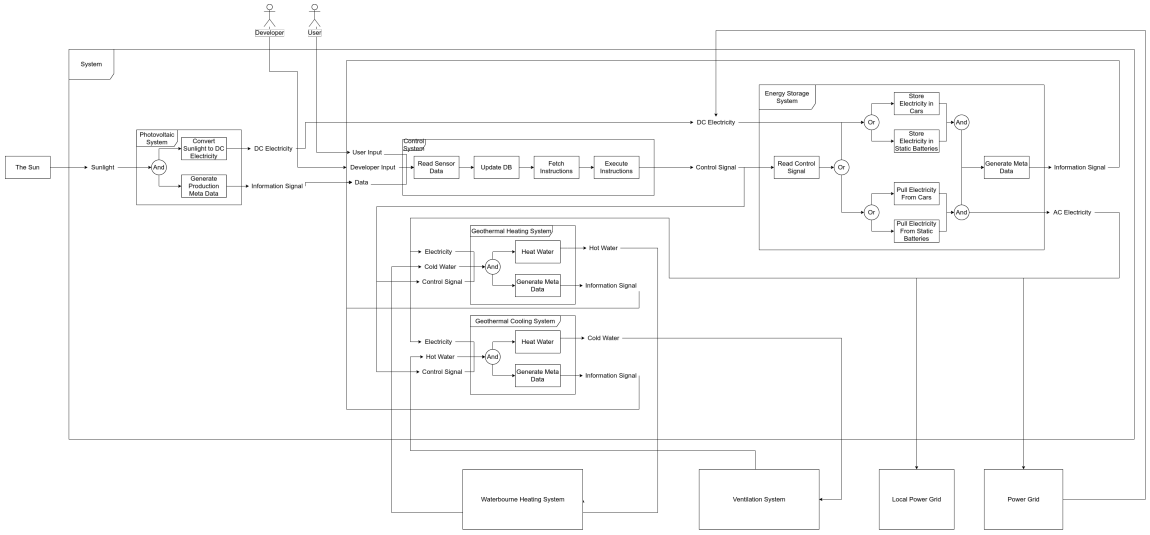


Figure 30: FFBD of the respective subsystems

Structure

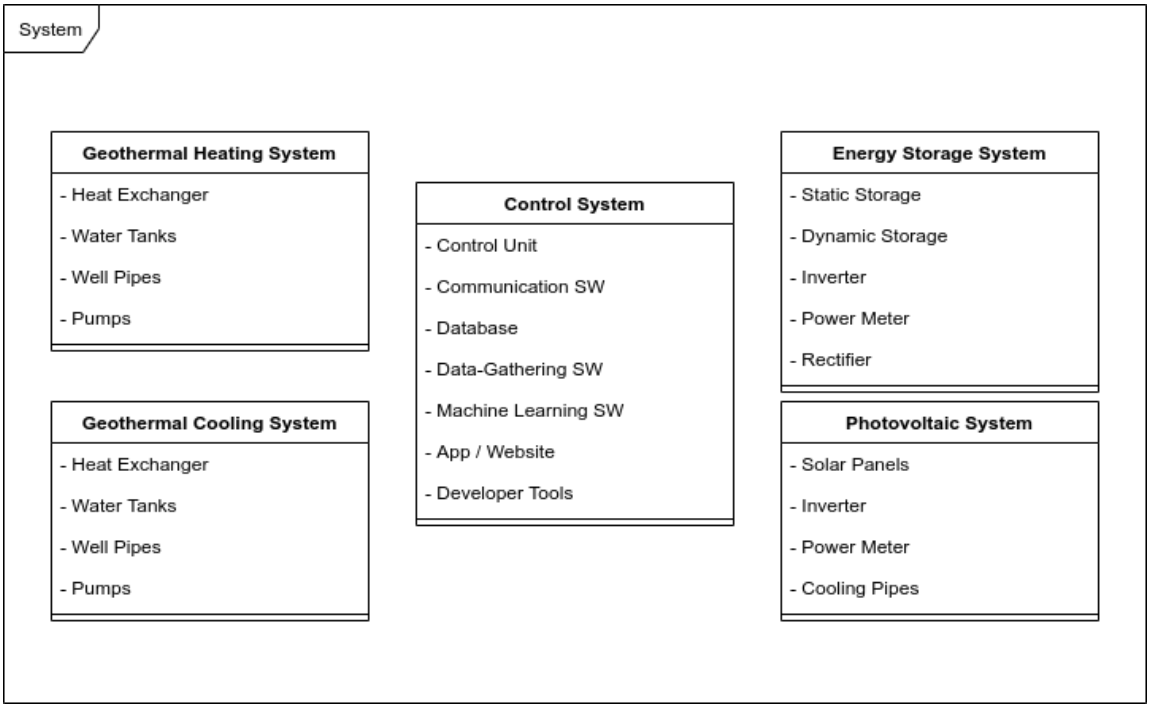


Figure 31: Class diagram showing the subsystems

In figure 31 we can see the structure of our system at this level of detail.

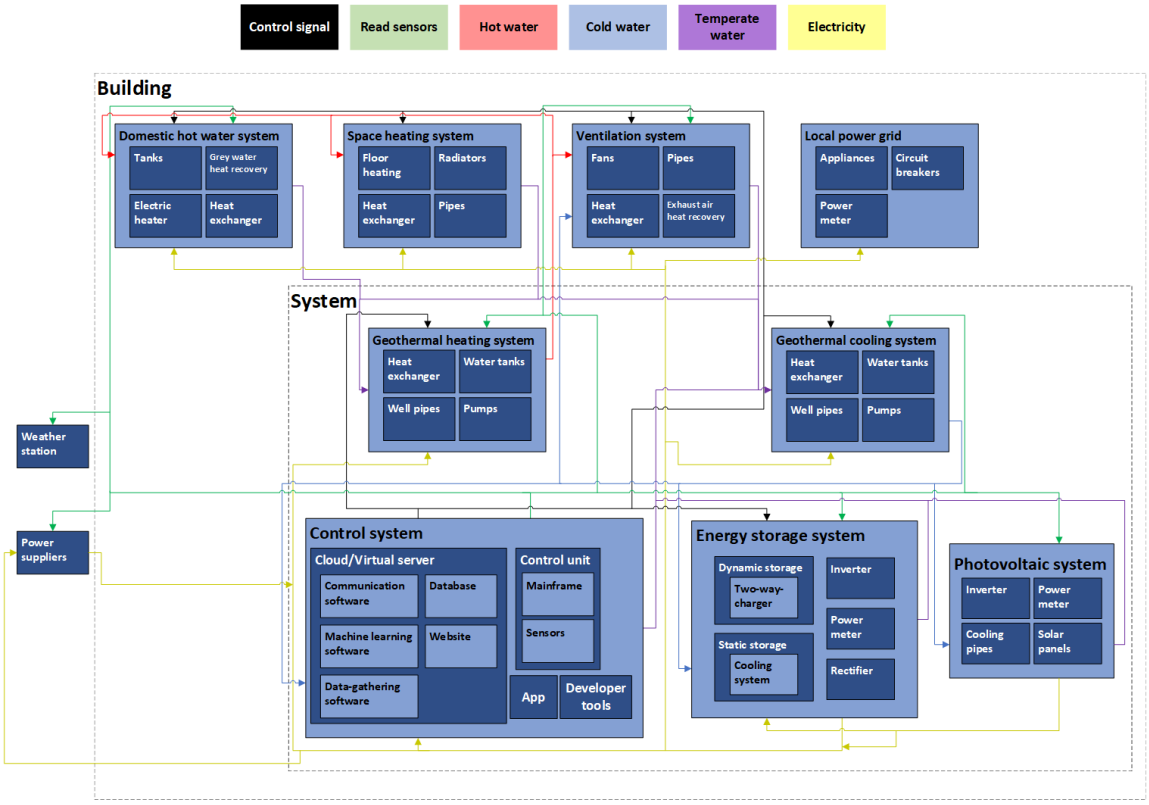


Figure 32: A more detailed structural model of our system and its context

In figure 32 we have taken the subsystems from figure 31 and put them in context with the building

Trade-Off

Control System

Our system needs to store large amounts of data and do a lot of computing. It will need continuous updates and adjustments to the machine learning algorithm. Buying a server with the capacity to store 20 years of data from day 1 is not a good economic option. We need to add vertical and horizonal scaling as the data amount and computing power need increases.

Each building complex will need a control unit in the physical building to communicate with the actual subsystems and even take some decisions in the case of internet loss.

But the main computing, data storage, and data gathering software does not need to be in the

physical building complex. So we made this PUGH matrix to compare the two options.

Criteria	Common cloud server	Individual building server
Response	2	5
Computing power / price	3	4
Modularity	5	3
Simplicity	4	3
Updating software	5	2
Hardware maintenance	5	1
Storage / price	5	3
Physical access	1	5
Safety	4	5
SUM	**34**	**31**

Figure 33: Pugh matrix for Control System

The main disadvantage with a server in each building is the need to update an increasing number of servers with the latest version of our software. And to travel between the locations to add hardware for performance increases. But it would have 99% of full functionality when the building complex loses contact with the internet. And the response time would be very low and not dependent on other companies/systems.

Recommendations

A common cloud server will be the best option for our system. It will save money and a better response time and full functionality without internet is not worth the complexity it adds.

Derived Requirements

In order to find the best buy and sell windows for power we got this data from NordPool [1]. And to maximize profits for our customers, we need to buy/sell as much as possible at the optimal hours.

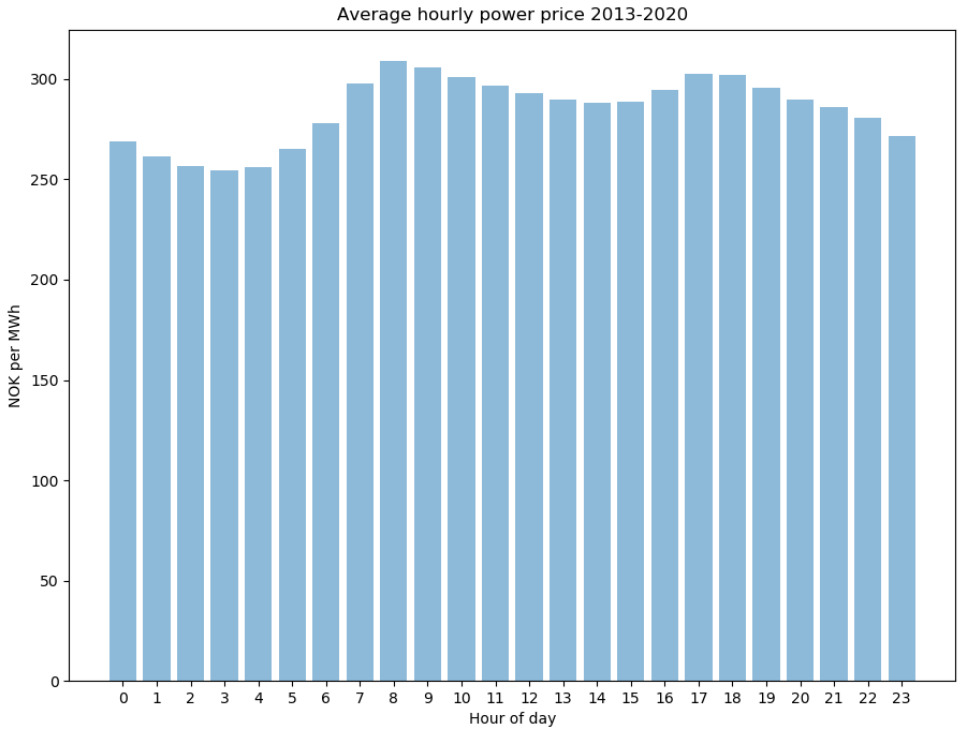


Figure 34: Average hourly power prices 2013-2020 (Source code on page 83)

Which gives us the requirements:

[Requirement 5.1.3.2]

[Requirement 5.1.3.3] HER MÅ VI SKRIVE NOE!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

REF TIER 2 requirement

Tier 2: Subsystems

Energy Storage System

The energy storage system is not the main focus of this project, however, we have decided to include a greater level of detail of it, in order to more easily dive into the specifics of the control system.

Behavior

Here we can see how we envision the allocation of electricity in our system. Even though most of the logic seen here actually lies with the control system, we find found it suitable to display the energy storage system with some of its context included. This is to help the reader understand which parameters the functions depend upon.

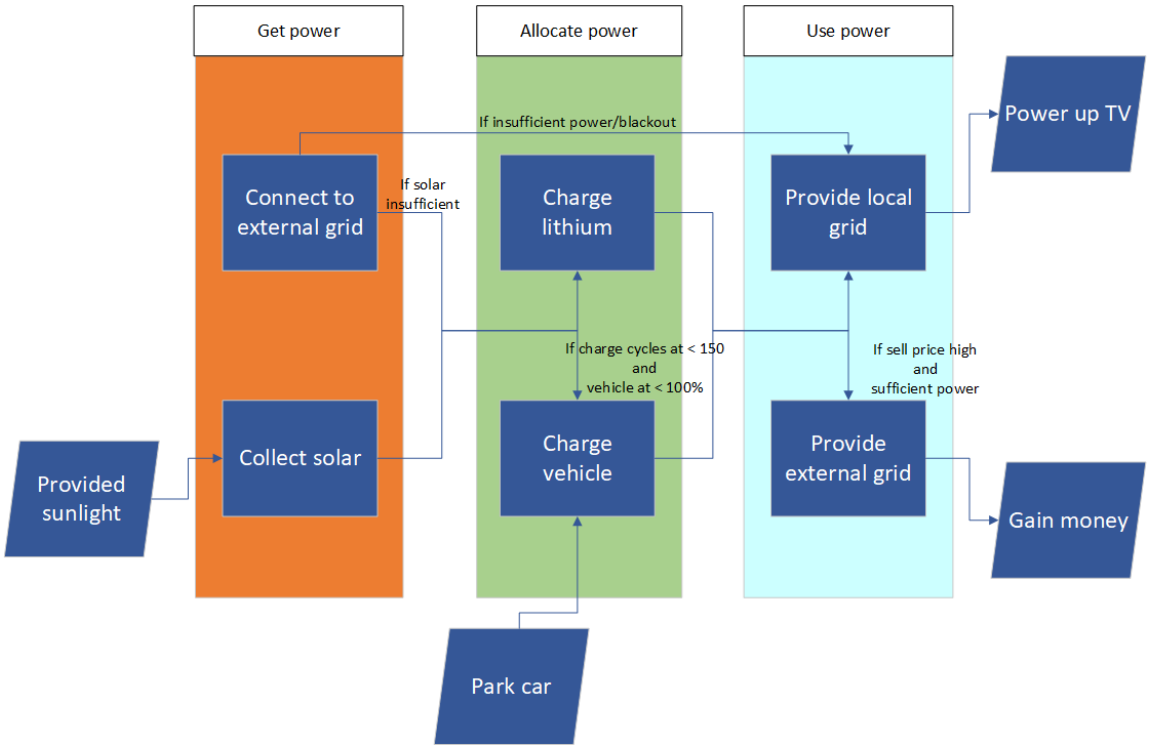


Figure 35: Behavior of the Energy Storage System

The main thing to draw from this model is that the energy storage system will receive electricity and store it in either the static batteries placed in the building, or in the vehicles owned by the

residents. It will also have to quickly allocate energy to power outlets, sell to grid or move electricity between the two storages.

Structure

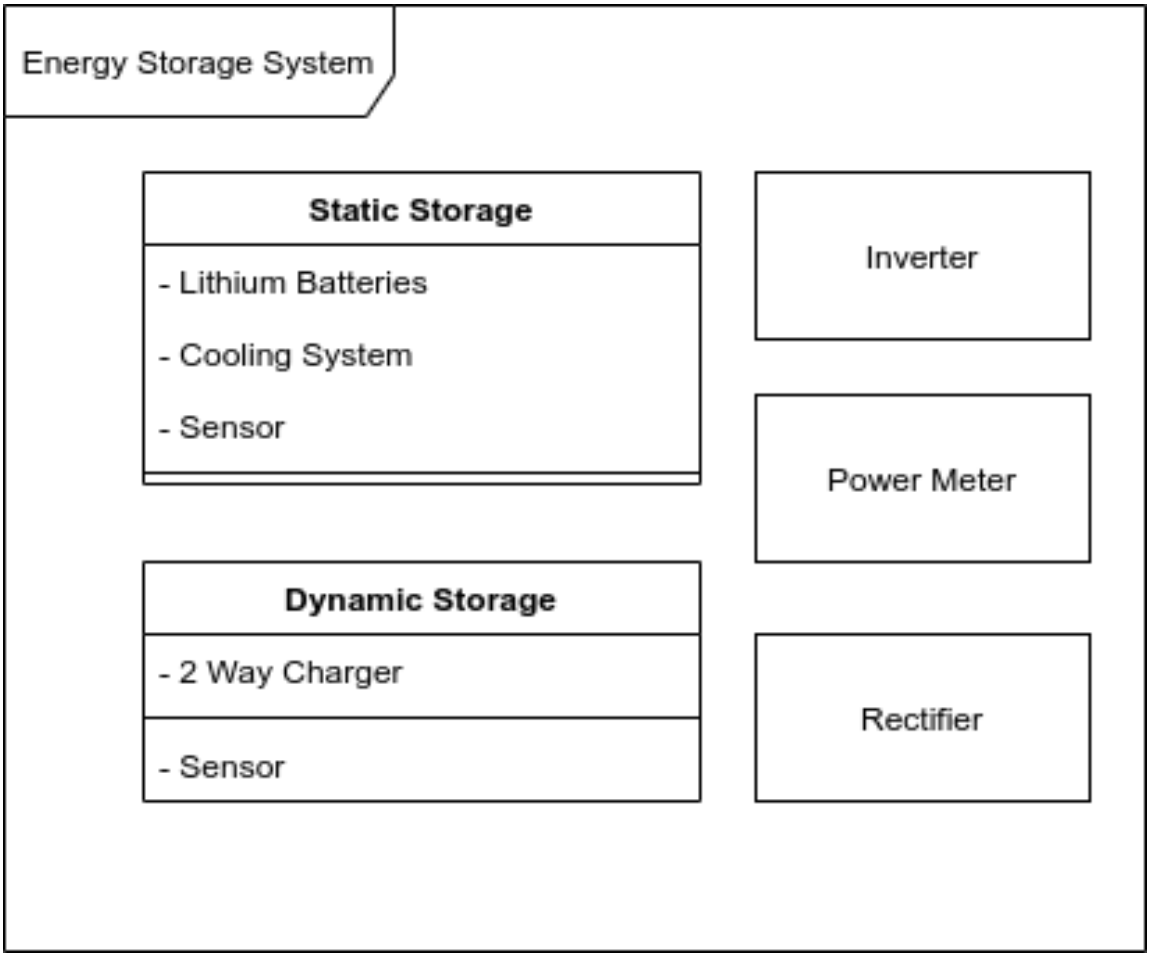


Figure 36: Class diagram of the Energy Storage System

The structure of the system is quite simple and will not be delved further into.

Control System

Operation

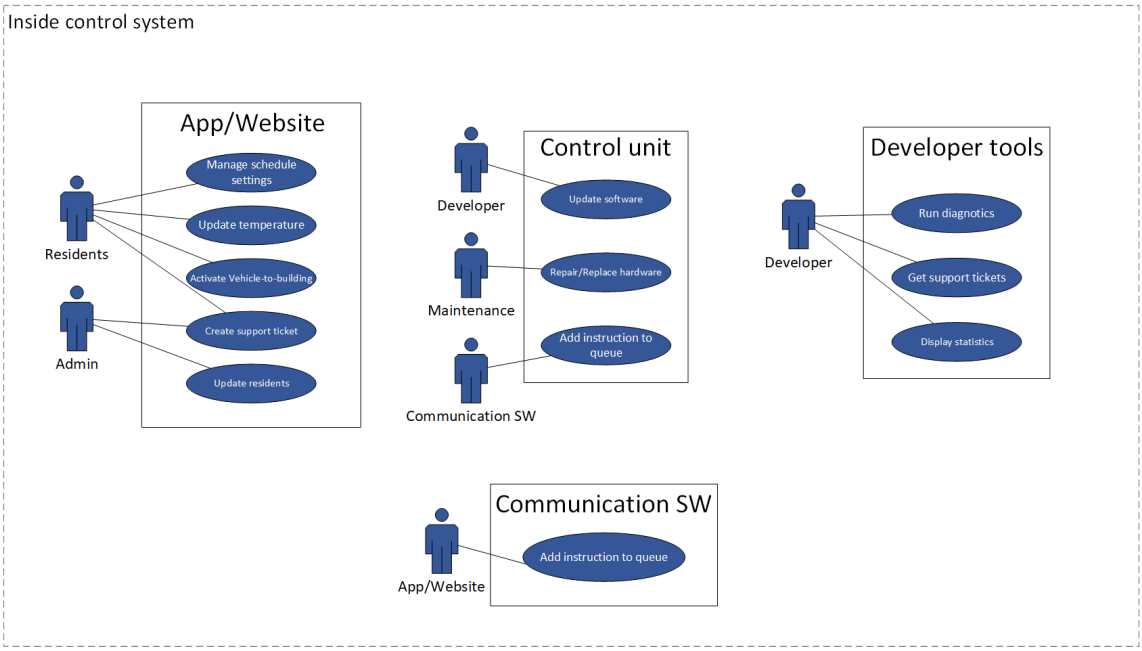


Figure 37: Use case diagram within the control system

These are the use cases we identified within the control system They are very abstract, but they helped us do the functional analysis for all the subsystems in the control system. We will get back to the structure of the components later, for now we will look at their behavior.

Behavior

Control Unit

The control unit is the component which will be installed in the building and will communicate with all the physical subsystems of our system as well as some systems that are not part of it.

In figure 38 we can see the functions it will need to execute:

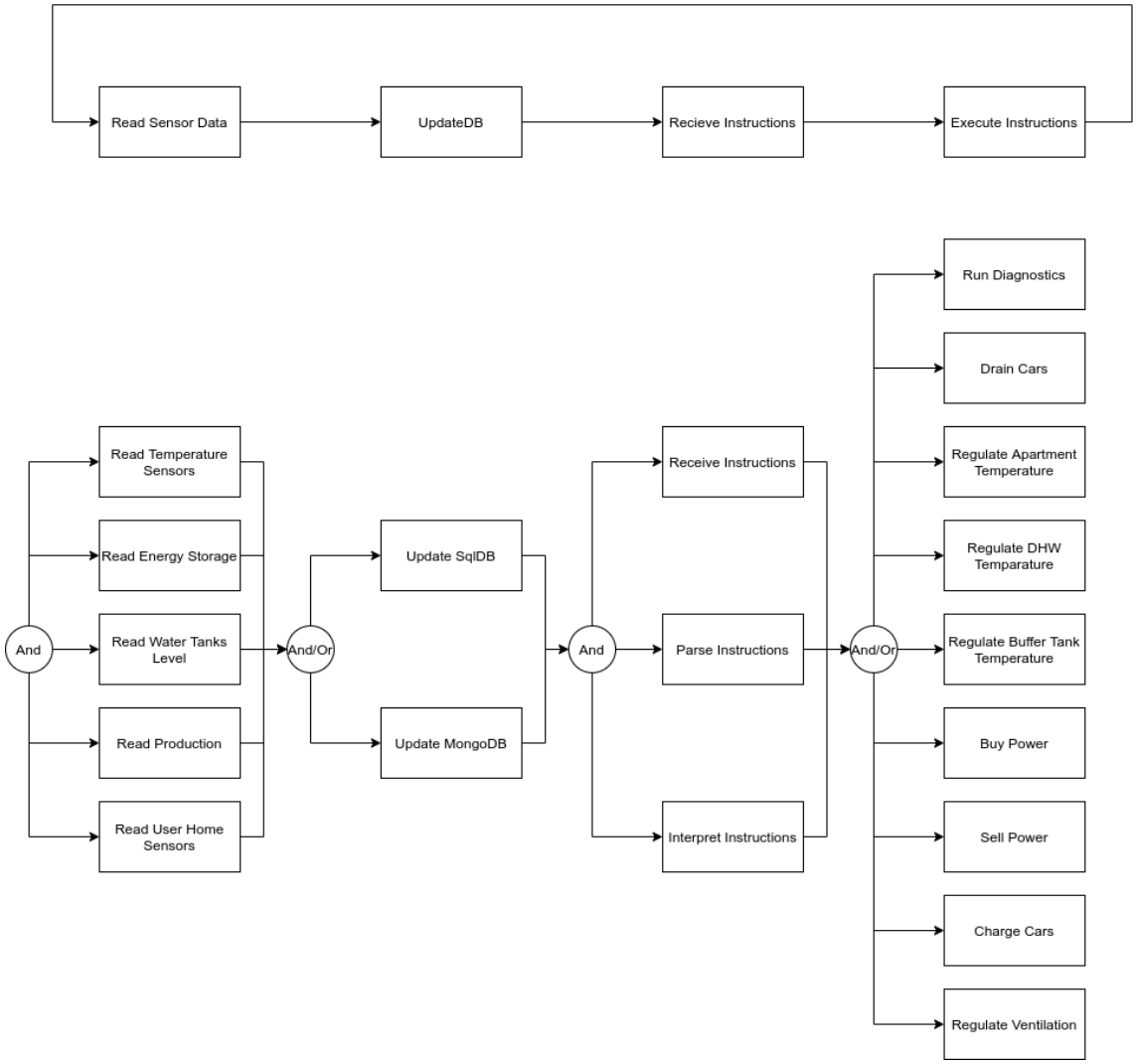


Figure 38: FFBD for the Control Unit

In the top of the figure we can see its main functions, and the respective functions are elaborated in the bottom. It will directly communicate with the sensors located in the buildings. This includes temperature sensors, the sensors in the physical components as well as sensing whether the residents are at home. All this data will have to be stored, so it will have to write it to the suitable database. Next, it will have to receive instructions from the MCU, and execute them. These instructions will determine the behavior of all parts of our system, as well as several others.

MCU (Main Communication Unit)

As mentioned in the last section, the MCU will send instructions to the control unit, and we can see the functions it will need in order to accomplish this in figure 39:

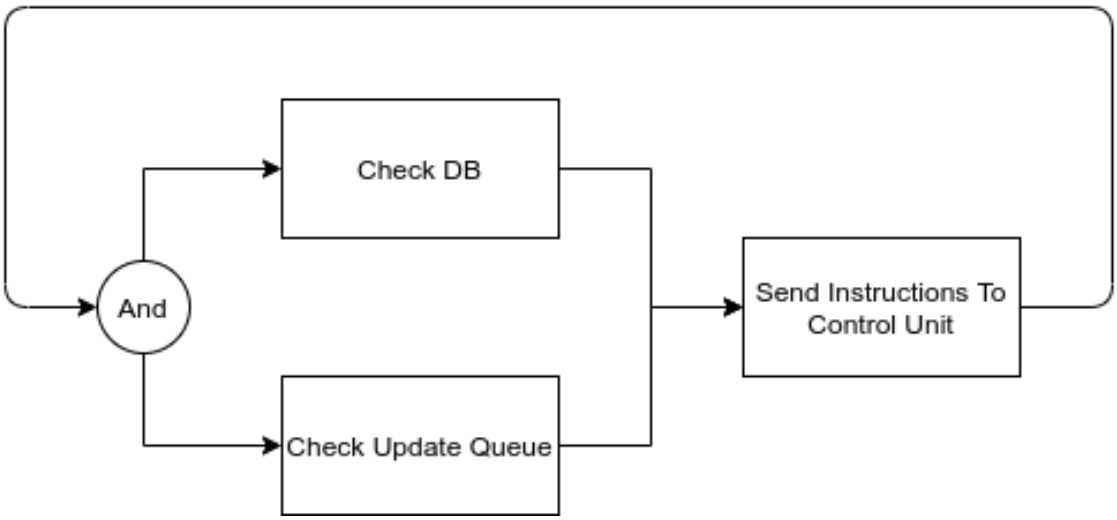


Figure 39: FFBD for the MCU

The MCU will gather instructions from the database, and the update-queue. In the database we will store the scheduled and long-term instructions for the system. However, this is not very suitable for real-time changes in wanted behavior, and these instructions will have to be gathered from the update-queue. This data-structure is mainly meant for unpredictable events, such as user inputs etc.

Databases

The basic functions we need in the databases are obvious. CRUD is common expression in database management, and it refers to the essential functions:

- Create
- Read
- Update
- Delete

However, the types of data we will need are quite different, so we need databases with different capabilities.

MySQL Often called the most successful software ever designed and is based on the Structured Query Language (SQL). MySQL stores its data in rows and columns and is optimal for relational databases. These are suitable for storing data that will have to be changed, as they are quite flexible. They are also easy to implement and offers strong data integrity. We will use SQL for storing user data, instructions, errors and many other things.

MongoDB The great majority of our data, however, will be stored in a MongoDB cluster. In order to make our system smart and able to make decisions, we will need tremendous amounts of data. MongoDB, whose name refers to humongous DB, is designed for this. The main difference in MongoDB, is that data is not stored in predetermined rows and columns in a table, but rather handled as collections of documents. The documents are JSON files which consist of key-value pairs. This means that we can store any attribute we will in every record of the table, but not every document needs the same attributes. Because of this, queries in MongoDB are much quicker than conventional SQL databases.

Data-Gathering Software

Now that we know how we will store the data; we will get into how we will obtain the data. As mentioned previously we will have probably hundreds of sensors in the buildings. However, we also need some external data. These data can be divided into two kinds:

Historical Data

We will need hourly historical solar, weather and power price data. Solar and weather are quite easy, as there exists free API's that provides us with accurate and extensive records. Power prices are a bit harder to acquire in the detail necessary. Nord Pool, the power exchange owned by the Nordic and Baltic countries do provide this, and for a yearly fee they allow access to rich amounts of data, so that cost will have to be factored in. This software will run only once per implementation and will therefore not be elaborated further on in this part.

Continuous Data

In addition to this we also need to continuously gather data. This includes getting weather forecasts and power prices. In figure X.X we can see how we have defined the behavior of the data-gathering software for storing continuous data:

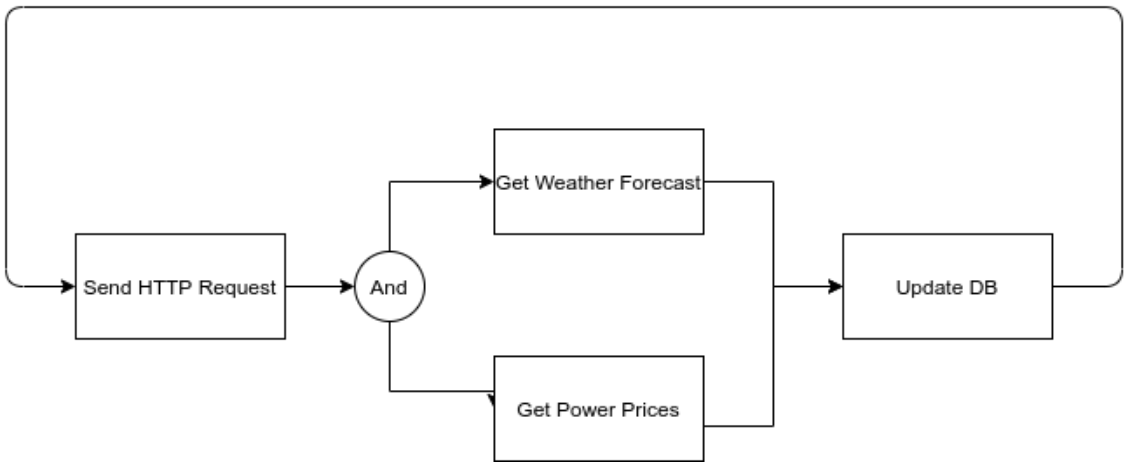


Figure 40: FFBD for the continuous data gathering software

This is a quite simple model, much like the software it depicts.

Machine Learning Software

The software for machine learning is essential in order to make our system smart and attractive and will be focused greatly on.

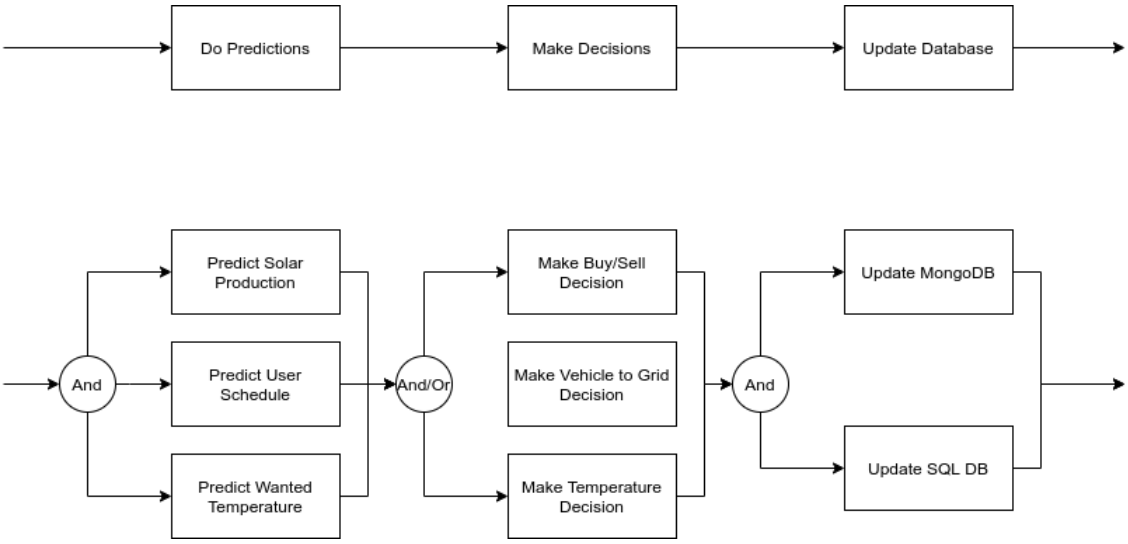


Figure 41: FFBD for the machine learning software

In the top model we see the abstract functions it will need to execute, and as we did with the Control Unit, we have also here found it necessary to unfold them. First, we will try and make predictions concerning the context of the system, how much power production we can expect and the needs and behavior of the user. In the next phase the software will have to make concrete decisions as to how the system should behave and operate. Lastly, we store these decisions as data and instructions in the databases.

Application / Website

Even though the system will be smart, there is no such thing as perfect. That is why we will need an opportunity to override the systems decisions. This is circular of course, because in order to be smart, the system will require large amounts of data about the users and their preferences. The smart device application and website will also be a platform for communicating information and statistics to the user.

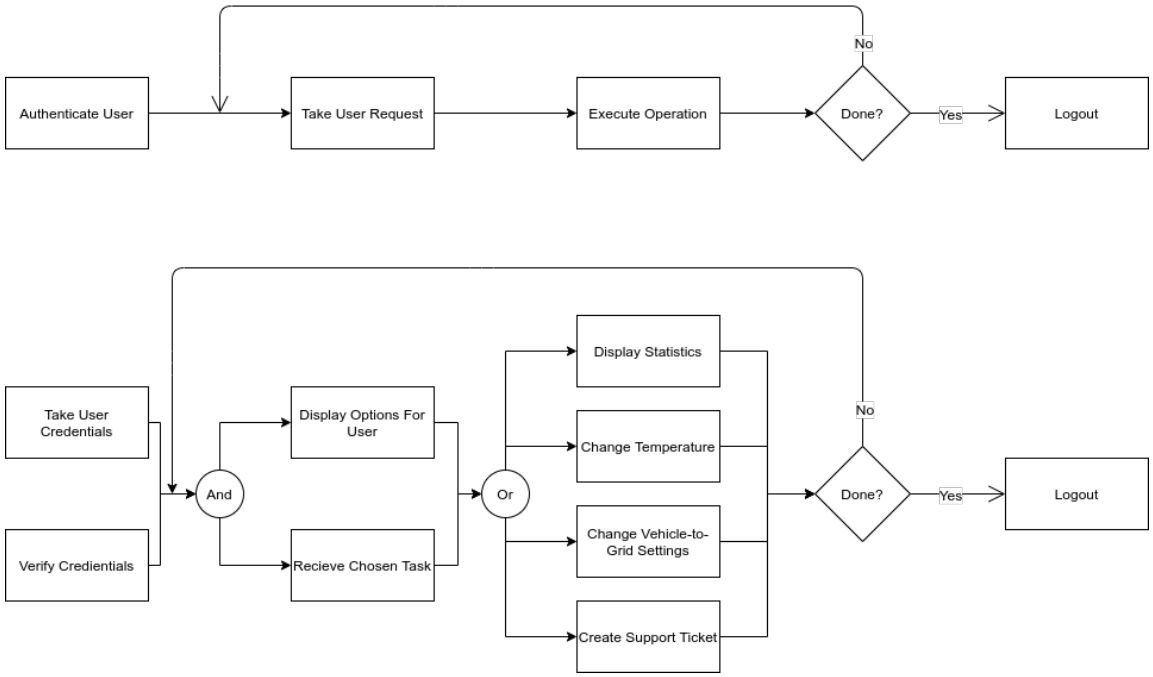


Figure 42: FFBD for the application/website

These are the main functions the application should offer the user. It should let them change the wanted temperature in their apartments, display statistics, configure the V2B system, as well as requesting support online.

Developer Tools

The Control System will need continuous monitoring, upgrading and maintenance. In order to do this effectively we have seen the need for a software which encapsulates tools for all of this.

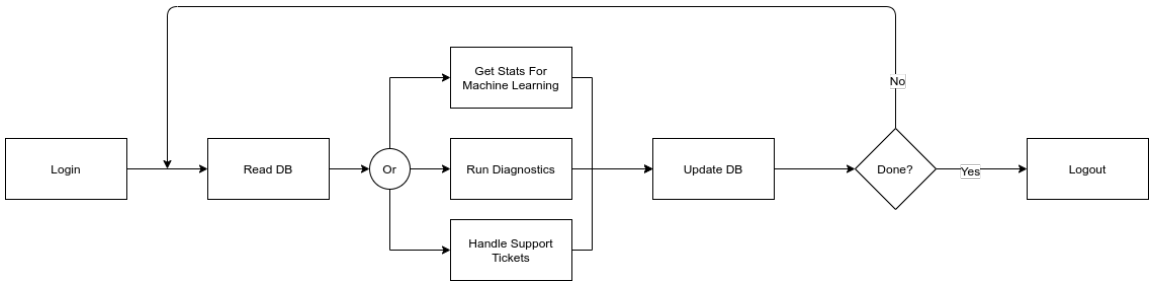


Figure 43: FFBD for the developer tools

Here we see among other things that the support tickets generated by the users, will be handled here. Now we have come full circle and will have a look at these subsystems will relate to each other.

Structure

After analyzing the functions of the subsystems within the control system, we were ready to look at the structure. In figure 44 we can see what components will make up the respective subsystems, where they will be located, and how they will be connected with each other.

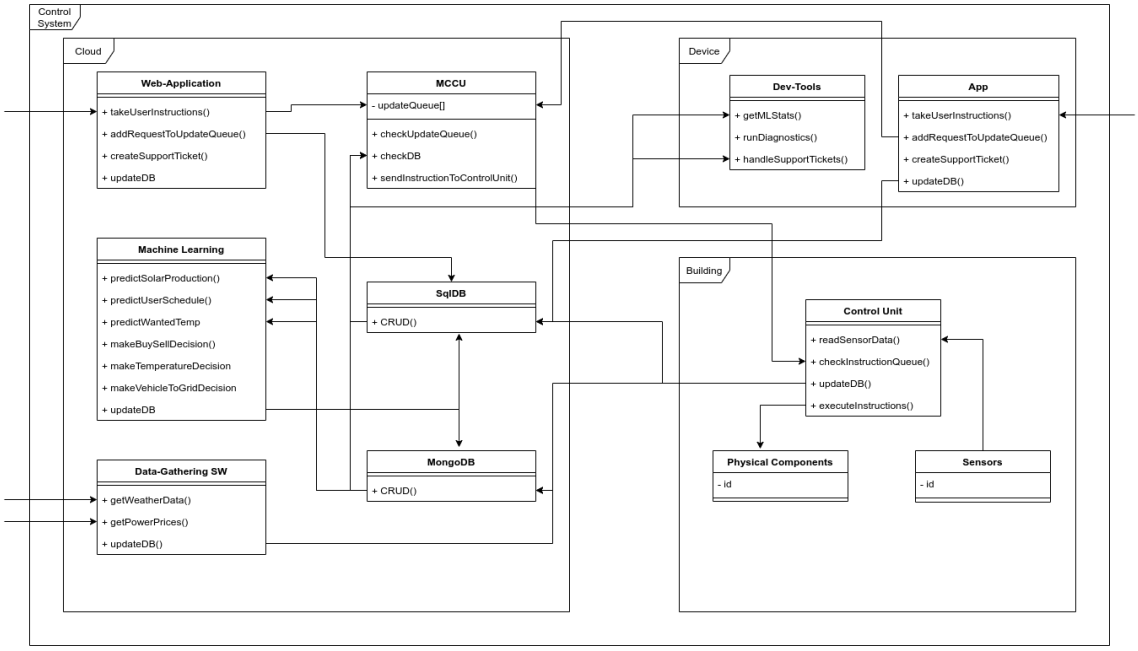


Figure 44: Class diagram for Control System

This is the deepest point of the system we have looked at, and we will now proceed with other aspects of our project.

Build and Test Plan

Verification

Verification plan is how we will use the data measured to verify the system matches the design requirements. “Did we build what we said we would”

To make sure that our system is ready for installation we set up this verification matrix where we set up the requirements and found test cases and methods of testing to confirm that the requirements are met.

INSERT IMPORTANT MAPPINGS!

Validation

Validation plan is how we will use the data measured to validate the system matches the business requirements. "Did we build what we need"

To confirm that the system operates according to its designed purpose we will have to put it through a validation test, we will be doing prospective validation which is validation before we install the system, and retrospective validation which is validation that is done while the system is running on location

Prospective Validation

- Is the system easy to learn? We will give the users access to user manuals and gain access to the user interface and get reports back whether learning the system is satisfactory.
- Does the system follow regulations? Invite inspectors for relevant government offices to confirm that the system is up to regulations.

Retrospective Validation

- Is the system regulating the apartment temperatures correctly? We will monitor the temperature provided to the apartments and verify that the temperatures matches the desire of the users.
- Is the system giving the correct information to the users? We will monitor the users input and confirm that the system fulfills those inputs.

Verification

In this section we have tried to test, simulate and/or calculate that the some of the key features for the system are achievable.

Production

To find photovoltaic production numbers we first guessed with a little help from some articles.

Then we sent out emails to local installers in the Oslo area and got a response containing the price and kWp effect of commonly used solar panels.

See the attached email at page 76

The building roof area is $1800m^2$ and to simulate what our photovoltaic production numbers would be we used the average solar data from 3 different satellites. This simulation calculates kWh produced using a static kWp effect solar panel. In reality this will vary based on temperature and wind speed. And Oslo has a good combination of solar radiation, wind speed and temperature.

For the angles we used the suggested optimal slope and azimuth from the website tool [13]. The roof panels will be mounted with a slope angle of 40°and an azimuth angle of 2°. Azimuth angle of 0°means it is facing directly south.

The solar panels mounted on the facade will use a slope angle of 90°. And for simplicity we used the azimuth angles for directly west, south and east. We did not have time to simulate which azimuth angle the entire building complex should be in for the best power production from all the walls put together.

In total the system will have $1350m^2$ of solar panels spread evenly across the west, south and east facing walls.

Production will of course have some variation throughout the year. But the amount of power we need to buy from the external power grid is less than 40% of our total power production. As we can see on the following graph our average self-sustain percent is ~64%.

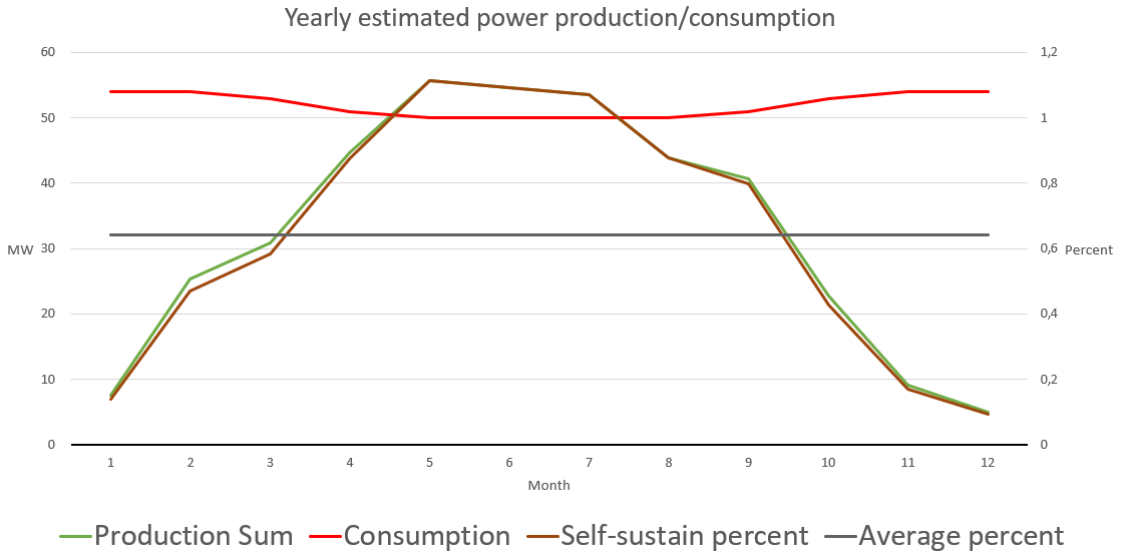


Figure 45: Graph of estimated yearly power production and consumption

Capacity for Storage

We estimate the average electric vehicle our residents will use to have a capacity of 75kWh. And the historical day with the highest solar radiation made our system produce 1.4MWh above the maximum sell limit. To make sure it truly is the worst-case scenario we also put consumption at 0. The challenge for our system is to not throw away energy. From 6 o'clock to 17 o'clock we need ∼19 electric vehicles to store all the energy. The problem is most people are at work from 9 to 15. We need ∼16 electric vehicles to store all the energy between 9 and 15.

For economic reasons we can only cover ∼3/8 of the total energy storage need with our static storage.

See the attached email at page 75

If the system has access to 0 electric vehicles, we will have to throw away 0.7MWh. Odds of 0 consumption, 0 electric vehicles and record-breaking solar radiation are low, but it can happen. These odds are acceptable when the alternative is to almost triple our static storage, we cannot cover all the scenarios.

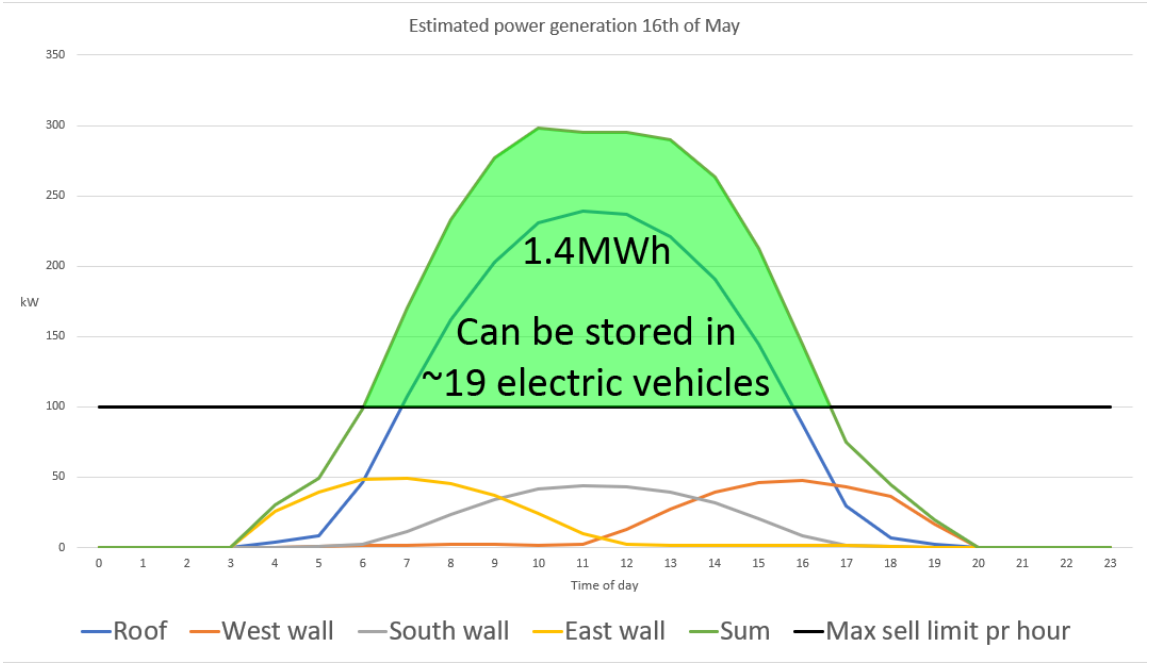


Figure 46: Graph of estimated 24 hour peak power production

Hopefully in the future Norway manages to update some current laws to make it easier for large

apartement complex to have their own energy production, without any need for energy storage. If we could "store" the energy by sending it to the external grid where someone else in Norway could use the power instead. And have the same amount removed from our power bill. It would be a much more economical and enviromental way to store energy compared to lithium batteries [2].

SOLIDWORKS MODELL Temperatur FINN PÅ BEDRE TITTEL

SKRIV NOE HER!!

SOLIDWORK Solar CElls weight FINN PÅ BEDRE TITTEL

SKRIV NOE HER!!

Prototypes

Machine Learning

As we've mentioned several times, machine learning will be heavily relied upon for making our system function properly and making it attractive. Because of this we found it necessary to verify the feasibility of this by prototyping and testing some of the key functions the machine learning software will have to perform.

The way we handle buying and selling of power are very important in order to not waste the energy we produce. We also want to buy and sell for the best possible price. To accomplish this, we need among other things to predict our solar panel power production, as well as predicting the power prices.

Predicting Solar Panel Power Production

n order to predict the solar panel power production, we needed historic weather and sun data. Luckily for us the European Commission provides a free API which offers data from the Photovoltaic Geographical Information System (PVGIS). This let us specify the location and direction of our solar panels, and outputted hourly data production data from 2005-2016. We stored the in our MongoDB cluster, one table for each of the walls, according to which cardinal direction they are facing, as well as one for the solar panels on the roof.

Weather data is easily available from many different sources, we chose to use the API of the Norwegian Meteorological Institute which provides extended information about cloud density and height. This also was stored in the cluster.

Python is a very good tool for prototyping machine learning and holds several libraries for different machine learning approaches. The ones we touched on work like this:

- Import the dataset
- Divide the data into two, one training set and one test set.
- Let the algorithm find a fitting model from the training set.
- Apply the model on the test set
- Calculate deviations and error-margins.

Next, we analyzed the results, and evaluated whether the approach was good enough. We tried a few approaches, among them linear regression, but eventually got good results with a decision tree regressor algorithm. These are for the south wall solar panels:

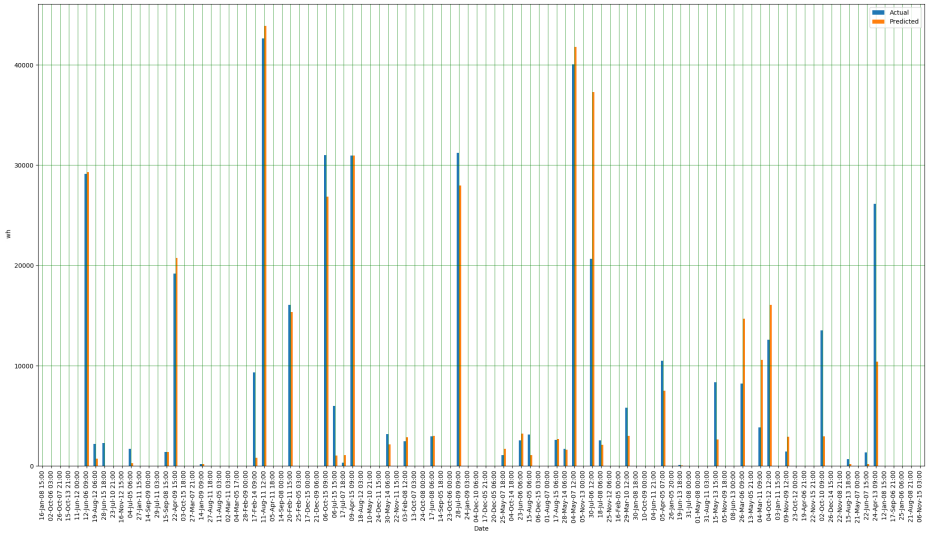


Figure 47: Machine learning prediction results for shuffled solar production

$$MeanAbsoluteError = 1.7kWh$$

The blue bars in this graph shows the actual power production during an hour, the orange bars show the predicted production and the number shows the accuracy of all the predictions, not just the data displayed in the model.

This looks a bit too good to be true, and in a way it is. When dividing the data into two, one can chose to shuffle the dataset, or to do it chronologically. Of course, it is easier for the algorithm to 'guess' on a value between a group of others, than if it had to predict the production for an extended period without as much information from the context. The chronological way is how it will work in a proper implementation, so it's crucial that this also produces acceptably results. These are the data we got for the panels on the south wall:

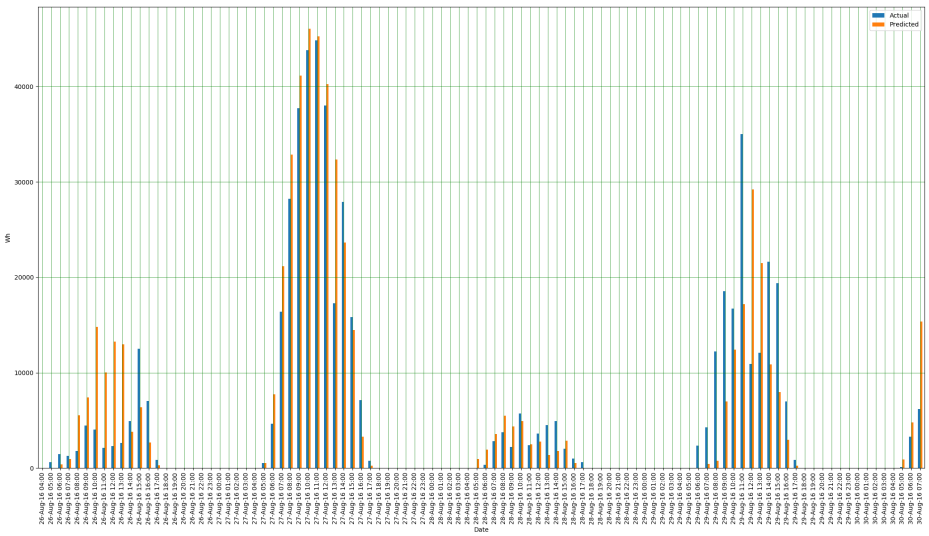


Figure 48: Machine learning prediction results for chronological solar production

$$MeanAbsoluteError = 2.3kWh$$

As expected, these results are much less accurate. However, we still find it viable, and these predictions will keep getting better as the data increases in size.

The source code is found on page 84

Predicting Power Prices

The price of electricity depends on supply and demand at any given moment. These two factors however are the results of a tremendous number of variables, which makes this task very difficult.

The first thing we looked at was the historical data:

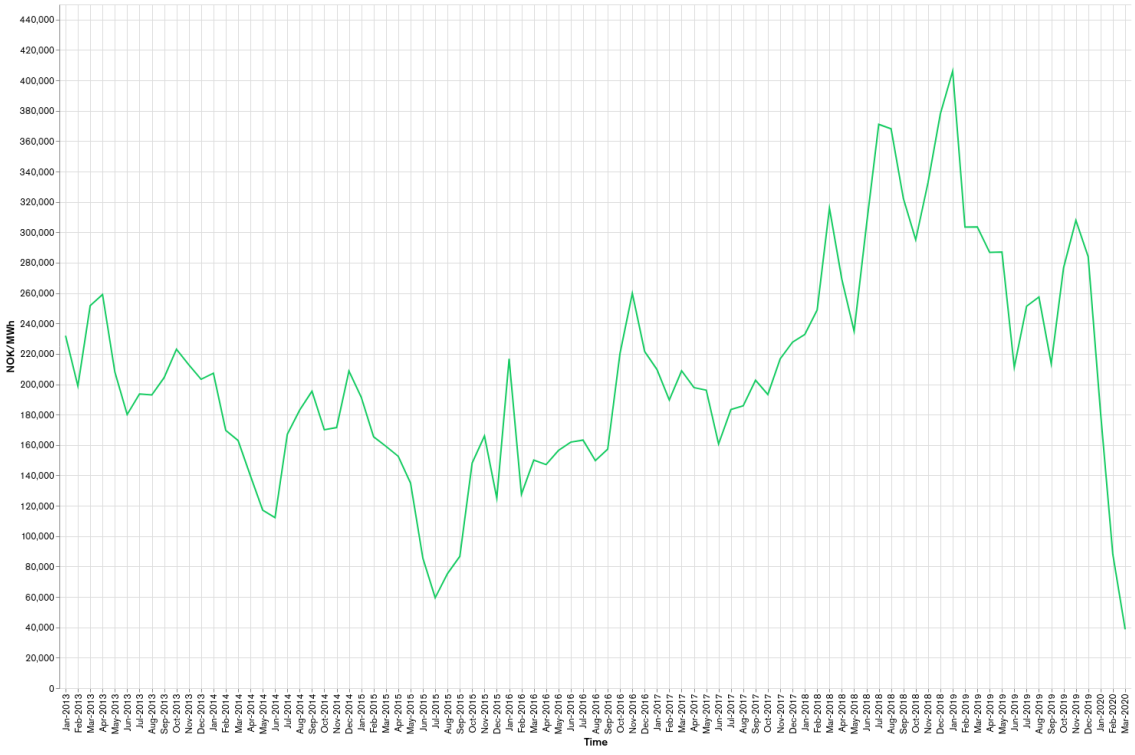


Figure 49: Power prices 2013-2020 in NOK/MWh

We gathered hourly historical data and analyzed them. The fact that they wary so greatly and abruptly brings further complexity to the problem. Still we fed the data to the same algorithm we used for predicting our production and got these results for a shuffled dataset:

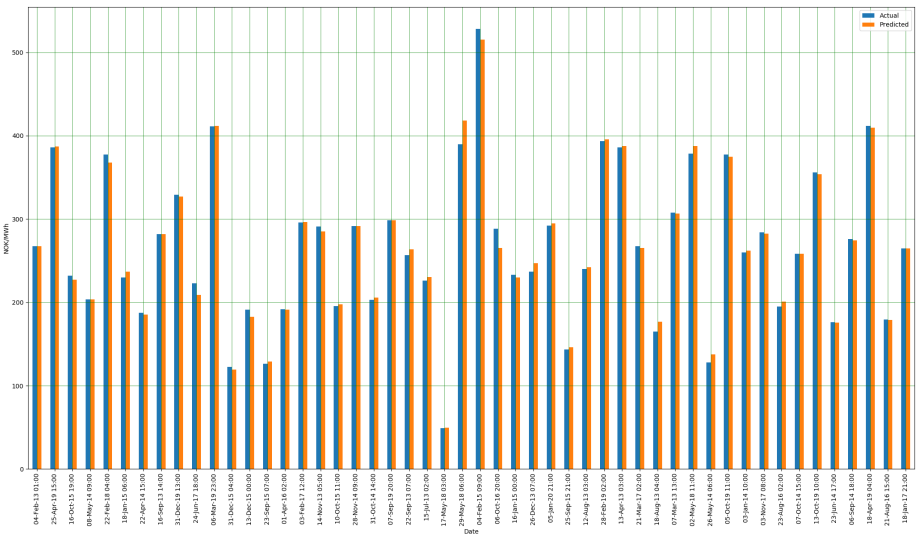


Figure 50: Machine learning prediction results for shuffled power prices

$$MeanAbsoluteError = 6.9NOK/MWh$$

The results are perfect, but they are still a victim of the same symptom we saw with shuffled lists with solar power, and to an even stronger degree. Still, this did seem promising with respect to chronological testing which ended up like this:

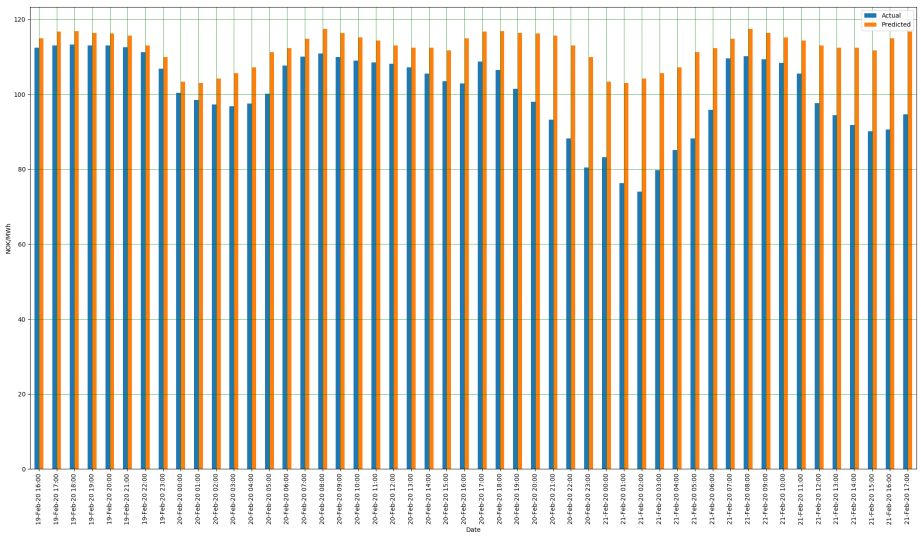


Figure 51: Machine learning prediction results for chronological power prices

$$MeanAbsoluteError = 33.6 NOK/MWh$$

The accuracy of the results dropped much more than what was the case with solar power, because the variables are too many, and we don't factor in anything but the historical data. Still, we concluded with that they are better than one would expect. We are confident that if more data, and more types of data are gathered, the results will increase in quality.

The source code is found on page 86

Web-Application

Residents should have an easy way to adjust their temperature and change their vehicle charge settings. And be given detailed statistics over their energy consumption the last 24 hours, week, month and year. We have focused on simplicity and provide both a website which will work on all modern browsers. And we will also compile the website code into an app for both Android and iOS.

We want our customers to have options to choose between website, app or both for their devices. And having our app on the respective app stores provides us with a tiny amount of marketing.

Which means we will only have one codebase to work on and update. But we will need to depend on other software for the actual conversions.

We have made a prototype of the "change temperature in apartment" function. And if we had more time we would continue with the "change electrical vehicle charge settings" and "energy consumption statistics".

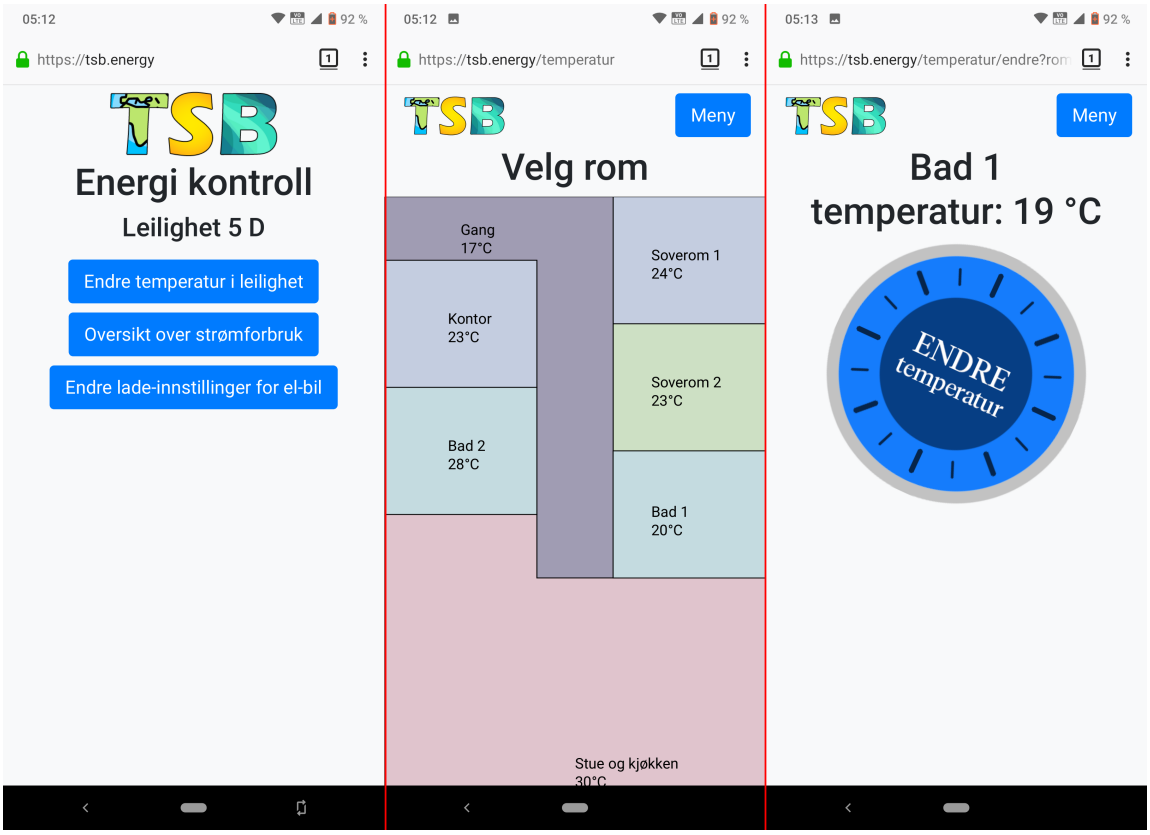


Figure 52: Screenshots of the website prototype

A video demonstration of the website/app interface can be seen at:
`https://drive.google.com/open?id=1qdpW8sEy_y_6BO9mgFI1ST6dhzjiOjjQ`

System Risk Analysis

We perform risk analysis for our system operation so one of these scenarios do not cause physical, emotional, psychological or social damage to our customer.

The risk analysis matrix shows how likely each scenario is and how much it will affect us or the stakeholders negatively. And the recommended action is what we can do to reduce the risk or the severity.

RISK	AREAS AFFECTED	SEVERITY	LIKELIHOOD	RISK IMPACT	RECOMMENDED ACTION(S)
Internet loss	Residents can not control temperature	LOW	MEDIUM	MEDIUM	Control unit has default values to fall back on for temperatures
Sensor failure	Temperature adjustment accuracy	LOW	MEDIUM	MEDIUM	Control unit try to hit set temperature by guessing at correct energy usage from historic data
Software bugs	Control system	LOW	MEDIUM	MEDIUM	Clean code, documentation, testing
Security breach	Control system	HIGH	LOW	MEDIUM	Encrypt or hash personal information

Figure 53: Risk matrix for system operation

Economics

Our business plan consists of selling our subsystems geothermal heat pump, vehicle-to-building, solar panels and lithium batteries as a one-time sale. And to provide a smart energy management service.

The subsystems we sell as a box are not crucial and are interchangeable if they match our subsystem requirements

The smart energy management service we provide has one physical mainframe in each building complex, temperature sensors inside and outside, and a proximity sensor in every apartment.

The mainframe and sensors provide essential data to our machine learning software, this data needs to be stored continually. And the machine learning software needs fine tuning and adjustments. Selling this as a service means less risk for us and the customer. Easier for us to maintain software with annual income, and the customer is guaranteed up to date software.

Our Budget

We made a budget to estimate our expenses and income. Unit sales will have a one-time expense/income while service will have annual expense/income. We will pay our engineers 1MNOK in the start and expect other expenses to be 100k annually.

For the most crucial and expensive sub-systems we tried to reach out to suppliers in the Oslo area and get a price from them. And some of them did generously respond to our queries. But for other sub-systems and atomic parts we simply guessed.

See the attached emails at pages: 74, 75, 76

The unit will us cost 7.7MNOK and the annual service will cost us 20k.

The unit will cost the customer 10MNOK and the annual service 50k.

	Year 1		Year 2		Year 3		Year 4		Year 5		Year 6	
	Half 1	Half 2	Half 1	Half 2	Half 1	Half 2	Half 1	Half 2	Half 1	Half 2	Half 1	Half 2
Investments	3 050k	3 050k	0	0	0	0	0	0	0	0	0	0
Sales												
Volume	0	0	1	1	2	2	3	3	4	4	4	4
Cumulative	0	0	1	2	4	6	9	12	16	20	24	28
Income												
Unit sales	0	0	10 000k	10 000k	20 000k	20 000k	30 000k	30 000k	40 000k	40 000k	40 000k	40 000k
Unit service	0	0	50k	100k	200k	300k	450k	600k	800k	1 000k	1 200k	1 400k
Expenses												
Salary	-3 000k	-3 000k	-3 000k	-3 000k	-4 000k	-4 000k	-4 000k	-4 000k	-5 000k	-5 000k	-5 000k	-5 000k
Other	-50k	-50k	-50k	-50k	-100k	-100k	-100k	-100k	-120k	-120k	-120k	-120k
Service												
Labour	0	0	-20k	-40k	-80k	-120k	-180k	-240k	-320k	-400k	-480k	-560k
Sales												
Material	0	0	-7 590k	-7 590k	-15 180k	-15 180k	-22 770k	-22 770k	-30 360k	-30 360k	-30 360k	-30 360k
Labour	0	0	-100k	-100k	-200k	-200k	-300k	-300k	-400k	-400k	-400k	-400k
Profit												
Half year	-3 050k	-3 050k	-710k	-680k	640k	700k	3 100k	3 190k	4 600k	4 720k	4 840k	4 960k
Cumulative	-3 050k	-6 100k	-6 810k	-7 490k	-6 850k	-6 150k	-3 050k	140k	4 740k	9 460k	14 300k	19 260k

Figure 54: Budget for our company

We expect to make a profit after 2 years. And be dept free from the 6 MNOK we loaned from investors after 3.5 years.

Our Cash Flow

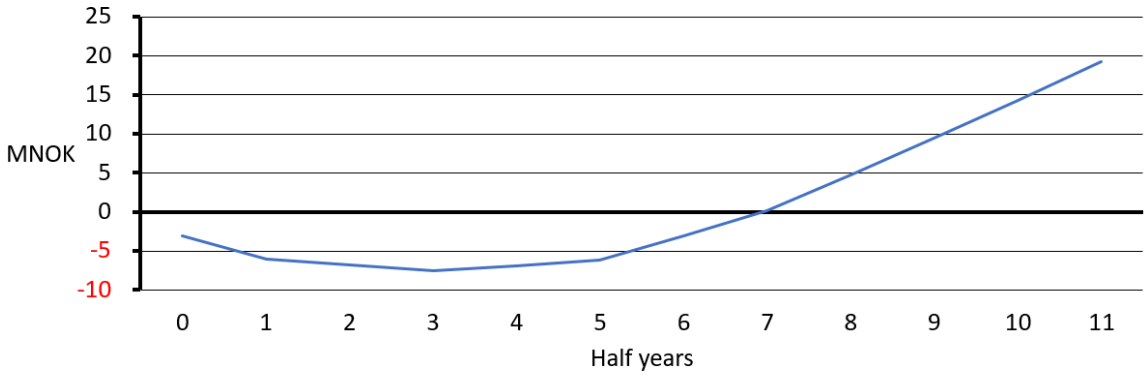


Figure 55: Cash flow graph for our company

Residents Budget

	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6	Year 7	Year 8	Year 9	Year 10
System purchase	-13 000k	0	0	0	0	-1 350k	0	0	0	0
Expense without system										
Power	-253k	-296k	-338k	-380k	-422k	-465k	-507k	-549k	-591k	-634k
System expenses	-4 000k	0	0	0	0	0	0	0	0	0
Cumulative	-4 253k	-5 098k	-5 774k	-6 534k	-7 379k	-8 308k	-9 322k	-10 420k	-11 603k	-12 870k
Costs with system										
Clean solar panels	-104k	-104k	-104k	-104k	-104k	-104k	-104k	-104k	-104k	-104k
Service-agreement	-50k	-50k	-50k	-50k	-50k	-50k	-50k	-50k	-50k	-50k
Income with system										
Surplus power sale	18k	20k	23k	26k	29k	32k	35k	38k	41k	44k
Profit										
Year	-13 136k	-134k	-131k	-128k	-125k	-1 472k	-119k	-116k	-113k	-110k
Cumulative	-13 136k	-13 270k	-13 401k	-13 528k	-13 653k	-15 125k	-15 244k	-15 360k	-15 473k	-15 583k

Figure 56: Budget for customer

Our system costs 10 MNOK, but the customer also needs to dig wells for the geothermal heat pumps. Which we estimate will cost 3 MNOK, putting the total at 13 MNOK.

Residents Cash Flow

Subsystem replacement schedule:

- Geothermal heat pump: 5 years
- Lithium batteries: 10 years
- Solar panels: 20 years
- Vehicle-to-building: 20 years

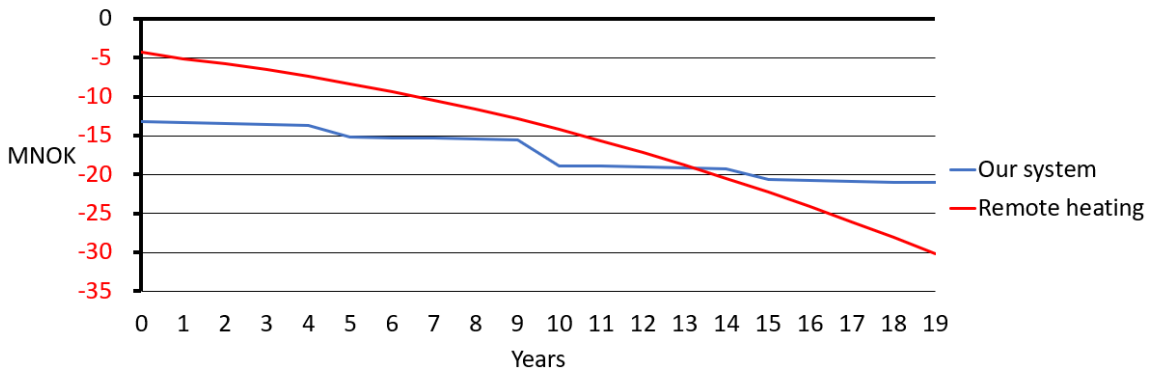


Figure 57: Cash flow graph for customer

ur remote heating cost estimation is very generous. And the residents will still end up saving money with our system after ∼13 years. We suspect remote heating is more expensive in reality.

Conclusion

About our system

Even though our system does not make a large multi-floor building 100% self-sustainable, it is still a huge help. It puts unused roof area to use, provides efficient heating and smart energy storage. Due to the urban environment we aimed our system at, it narrowed the opportunities for energy harvesting.

However, since most people live in these urban environments, we feel that our system will make a much bigger impact than for instance a 100% renewable house in the countryside. The opportunities

for scaling a smart energy system is much more important than the actual percentage of renewable energy achieved.

One of the problems we see with large buildings having their own power production is the need for energy storage. If the regulations do not let you sell all excess power, there will be a lot of unnecessary energy waste. Producing lithium batteries for all large and small buildings is not a good long-term solution as the world population grows. In a country like Norway where virtually all power comes from renewable sources, it is better to rely on the power grid.

What did we learn

- Working in a team
- How to attack large problems
- View the system from different stakeholders point of view
- Modelling, a picture says a thousand words
- Communicate ideas and concepts (T-shape)
- The importance of time-boxing
- Separating problem from solution
- Synthetic vs analytic thinking
- How to cooperate using digital platforms
- There is always another interface
- Reading and understanding technical datasheets

What could be improved

It was difficult to scope the problem, and we should have put more focus on this earlier. This would have prevented us from spending time designing a building.

The focus of this assignment was to learn system engineering, and we were told not to focus too much on numbers. But poor research can lead to a poor product solution. The system is only as good as its worst sub-system.

The energy consumption of the building is a crucial starting point. The entire system is dependant on how much power we need. Research into this area should have had a larger focus early on, to make sure we were solving the correct problem.

References

[1] Nordpool Group ,. https://www.nordpoolgroup.com/. Accessed: March-20.

[2] Plusskunder, nve. https://www.nve.no/reguleringsmyndigheten/nettjenester/nettleie/tariffer-for-produksjon/plusskunder/. Accessed: March-20.

[3] Rooftop Wind Turbines: Are They Worthwhile? , engineering.com. https://www.engineering.com/ElectronicsDesign/ElectronicsDesignArticles/ArticleID/9556/Rooftop-Wind-Turbines-Are-They-Worthwhile.aspx. Accessed: March-20.

[4] Siragrunnen vindpark , multiconsult. https://www.multiconsult.no/prosjekter/siragrunnen-vindpark/. Accessed: March-20.

[5] Dette bruker du strømmen til , smartepenger.no. https://www.smartepenger.no/boligokonomi/2438-dette-bruker-du-strommen-til. Accessed: March-20.

[6] Elevator energy calculator, thyssenkrupp. https://www.thyssenkruppelevator.com/tools/energy-calculator/. Accessed: March-20.

[7] Fremtidens bolig, enova. https://www.enova.no/upload_images/42B0BD408ABB4271A1CDEABF9C402A6E.pdf. Accessed: March-20.

[8] Ikke nok sol til solceller i Norge? , fremtiden.no. https://www.framtiden.no/myteknusing/ikke-nok-sol-til-solceller-i-norge.html. Accessed: March-20.

[9] Kjørelengder, ssb. https://www.ssb.no/transport-og-reiseliv/statistikker/klreg. Accessed: March-20.

[10] Krav til energieffektivitet TEK17 , direktoratet for byggkvalitet. https://dibk.no/byggereglene/byggteknisk-forskrift-tek17/14/14-2/. Accessed: March-20.

[11] Krav til passivhus NS3700 , tekna.no. https://www.tekna.no/fag-og-nettverk/bygg-og-anlegg/byggbloggen/krav-til-passivhus. Accessed: March-20.

[12] Norwegian household power consumption, ssb. https://www.ssb.no/energi-og-industri/artikler-og-publikasjoner/vi-bruker-mindre-strom-hjemme. Accessed: March-20.

[13] Photovoltaic Geographical Information System , european commision. https://re.jrc.ec.europa.eu/pvg_tools/en/tools.html. Accessed: March-20.

[14] Samlet energibruk , nve. https://www.nve.no/energibruk-effektivisering-og-teknologier/energibruk/samlet-energibruk/?ref=mainmenu. Accessed: March-20.

[15] Strømforbruk for elbil, fjordkraft. https://www.fjordkraft.no/strom/stromforbruk/elbil/. Accessed: March-20.

[16] Strømforbruk og strømsparing, hafslund strøm. https://www.hafslundstrom.no/strom/privat/stromforbruk/2025. Accessed: March-20.

[17] Veiledning TEK17 , direktoratet for byggkvalitet. https://dibk.no/byggereglene/byggteknisk-forskrift-tek17/. Accessed: March-20.

[18] David W. Oliver, Timothy P. Kelliher, and James G Keegan. *Engineering Complex Systems, With Models and Objects.* McGraw-Hill, 1997.

[19] Richard Stevens, Peter Brook, Ken Jackson, and Stuart Arnold. *Systems Engineering, Coping with Complexity.* Pearson, 1998.

Appendices

A Emails

A.1 Geothermal heat pump supplier email

Tarald Vestbøstad

From: Øivind Botnevik <ob@abkqviller.no>
Sent: mandag 9. mars 2020 15.15
To: Tarald Vestbøstad
Subject: NIBE

Hei,
Budsjettpris for 9 stk. F1345-60 er 1 350 000,-.

Med vennlig hilsen

Øyvind Botnevik
Leadskoordinator

ob@abkqviller.no
+47 53 00 72 61 | +47 970 84 211

ABK-Qviller AS | Region Bergen
Hardangerveien 72, seksjon 21, 5224 Nesttun
abkqviller.no | +47 23 17 05 20

Meld deg på årets arrangement for våre forhandlere:

Kundehenvendelse fra kontaktskjema på NIBE.no

E-post: 141462@usn.no

Hei NIBE
Mitt navn er Tarald og jeg studerer data ingeniør ved USN.
Vi har et prosjekt som handler om smarte bygninger, og vår gruppe trenger en pris estimering på en varmepumpe.

Vi er veldig takknemlig hvis dere kunne hjelpe oss med en ca. pris for en bedrift å kjøpe inn ni stk NIBE F1345-60?

Mvh
Tarald Vestbøstad

1

A.2 Lithium battery supplier email

Tarald Vestbøstad

From: Kjell Are Sangvik <kas@skanbatt.no>
Sent: tirsdag 10. mars 2020 07.06
To: Tarald Vestbøstad
Subject: Re: Skanbatt lithiumbatteri pris estimat for skoleprosjekt

Hei igjen.

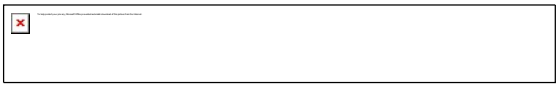
Uten å ha noen videre spesifisering på hvordan denne batteribanken skal være, så vil prisen være ca 1,8 mill eks mva.
Dette er ut ifra dagens dollar kurs.
Dersom dette er noe dere tenker å gå videre med, trenger jeg litt mer opplysninger om denne batteribankens behov.

Med vennlig hilsen / Best regards

Skandinavisk Batteriimport AS

Kjell Are Sangvik
Tlf:+47 47245325
Web: www.skanbatt.no

Supportforum: https://www.facebook.com/groups/SkanbattForum/



man. 9. mar. 2020 kl. 10:50 skrev Kjell Are Sangvik <kas@skanbatt.no>:

> Hei.
>
> Takk for deres henvendelse.
> Jeg skal sjekke med leverandør på en slik stor pakke, hva dette kan komme på.
>
> Kommer med en tilbakemelding når jeg vet noe.
>
> Med vennlig hilsen / Best regards
>
> **Skandinavisk Batteriimport AS**

A.3 Solar panel supplier email

Gmail **Vegard Bernhardsen <vegard.g.bernhardsen@gmail.com>**

Nøkkeltall
2 e-poster

Torgersen, Morten <Morten.Torgersen@smartly.no> 17. februar 2020 kl. 14:09
Til: "vegard.g.bernhardsen@gmail.com" <vegard.g.bernhardsen@gmail.com>
Kopi: Support Smartly <support@smartly.no>

Hei Vegard

Et anlegg som dekker 6400kvm, vil normalt sett gi følgende tall:

- Med ca 160 Wp/kvm gir dette 1 024 000Wp, altså 1 024kWp
- Med en markedspris på ca 9kr/ Wp vil et slikt anlegg koste ca 9,2 mill NOK.

Men det bør være fullt mulig å få det rimeligere med en god tilbudsrunde, så 7-8 mill bør være innen rekkevidde.

Dette er veldig omtrentlige tall, men kan være greit å bruke som eksempel.

Vennlig hilsen

Morten Torgersen

Prosjektleder - Fagansvarlig

Smartly AS

Mobil: +47 475 09 187

E-post: morten.torgersen@lyse.no

www.smarly.no

smartly

Vegard Bernhardsen <vegard.g.bernhardsen@gmail.com> 17. februar 2020 kl. 14:36
Til: "Torgersen, Morten" <Morten.Torgersen@smartly.no>

Hei
tusen takk for raskt svar, det er til stor hjelp.

hilsen
Vegard Bernhardsen

B Code

B.1 Historical Weather Web-Crawler

```
package frostapi;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.time.YearMonth;
import org.apache.commons.codec.binary.Base64;
import org.jsoup.Connection;
import org.jsoup.Jsoup;

public class FrostAPI {

    private final static String STASJON = "18700";
    private final static String[] DATAELEMENTS = {"air_temperature",
        "cloud_area_fraction",
        "cloud_area_fraction1",
        "cloud_area_fraction2",
        "cloud_area_fraction3",
        "cloud_area_fraction4",
        "surface_air_pressure",
        "wind_speed",
        "wind_speed_of_gust",
        "dew_point_temperature",
        "wind_from_direction",
        "relative_humidity",
        "precipitation_amount",
        "weather_type",
        "weather_type_automatic"};
    private final static String AUTH = "REDACTED";
    private final static String BASE64AUTH = new
         String(Base64.encodeBase64(AUTH.getBytes()));
    private static BufferedWriter bw;
    private static BufferedWriter logWriter;

    public static void main(String[] args) throws Exception {
        logWriter = new BufferedWriter(new FileWriter(new File("errors")));
        bw = new BufferedWriter(new FileWriter(new File("fixedData.json")));
        bw.write("{");
        bw.newLine();
        for (int i = 2005; i <= 2016; i++) {
            loopYear(i);
        }
        bw.newLine();
        bw.write("}");
```

```
    bw.close();
    logWriter.close();

}

private static void loopYear(int year) throws Exception {
    for (int i = 1; i <= 12; i++) {
        YearMonth yearMonthObject = YearMonth.of(year, i);
        int daysInMonth = yearMonthObject.lengthOfMonth();
        for (int j = 1; j <= daysInMonth; j++) {
            sendRequest(year, i, j);
        }
    }

}

private static void sendRequest(int year, int month, int day) throws Exception {
    String concat = ",";
    for (int i = 0; i < 24; i++) {
        try {
            Connection.Response get =
                 Jsoup.connect("https://frost.met.no/observations/v0.jsonld?sources=sn"
                    + STASJON + "&referencetime=" + fullNumber(year) + "-"
                    + fullNumber(month) + "-" + fullNumber(day) + "T" + fullNumber(i)
                    + "&elements=" + constructElements())
                    .header("authorization", "Basic " + BASE64AUTH)
                    .method(Connection.Method.GET)
                    .ignoreContentType(true)
                    .execute();

            if (i != 0) {
                concat += ",";
            }
            concat += "\"" + fullNumber(year) + fullNumber(month)
                    + fullNumber(day) + fullNumber(i) + "\" :" + get.body();
        } catch (IOException e) {
            logWriter.write(fullNumber(year) + fullNumber(month)
                    + fullNumber(day) + fullNumber(i));
            logWriter.newLine();
            logWriter.write(e.toString());
            logWriter.newLine();
            logWriter.write(e.getMessage());
            logWriter.newLine();
        }
    }
    bw.write(concat);
}

private static String fullNumber(int number) {
    if (number < 10) {
```

```
            return "0" + number;
        } else {
            return Integer.toString(number);
        }
    }

    private static String constructElements() {
        String elements = "";
        for (int i = 0; i < DATAELEMENTS.length; i++) {
            if (i != 0) {
                elements += ",";
            }
            String dataElement = DATAELEMENTS[i];
            elements += dataElement;
        }
        return elements;
    }

}
```

B.2 MongoDB Filler

```
#include <iostream>
#include <fstream>
#include <string>
#include <bsoncxx/builder/stream/document.hpp>
#include <bsoncxx/json.hpp>
#include <mongocxx/client.hpp>
#include <mongocxx/instance.hpp>
#include "boost/property_tree/ptree.hpp"
#include "boost/property_tree/json_parser.hpp"
#include "boost/date_time/posix_time/posix_time.hpp"
#include <boost/foreach.hpp>
#include <thread>
#include <vector>
#include <functional>
#include <algorithm>
#include "power.h"
#include <boost/algorithm/string/replace.hpp>

using boost::property_tree::ptree;
using boost::posix_time::ptime;
using boost::posix_time::from_iso_string;

bool fillSunWall(std::string, mongocxx::client&);
bool fillWeather(mongocxx::client&);
bool parseSun();
```

```
bool fillPowerPrices(mongocxx::client&);

std::string powerPath = "data/PowerData/El";
std::string powerExt = ".csv";

std::string weatherPath = "data/WeatherData/data2.json";
std::string sunPath = "data/SolarData/";
std::string jsonExt = ".json";


int main(int, char**)
{
    mongocxx::instance inst{};
    mongocxx::client conn{
        mongocxx::uri{
            "mongodb+srv://kent:<Password>@terrasolaris-robmp.azure.
      mongodb.net/test?retryWrites=true&w=majority"}};

    fillSunWall("roof", conn);
    fillSunWall("south", conn);
    fillSunWall("west", conn);
    fillSunWall("east", conn);
    fillWeather(conn);
    fillPowerPrices(conn);

    return 0;
}

bool fillSunWall(std::string direction, mongocxx::client& conn)
{
    for(int i = 1; i <= 3; i++)
    {
        std::cout << "Document: " << direction + std::to_string(i) << "\n";
        std::ifstream ist(sunPath + direction + std::to_string(i) + jsonExt);
        ptree pt;
        read_json(ist, pt);

        pt = pt.get_child("outputs").get_child("hourly");

        for (auto & array_element : pt) {

            std::string time = array_element.second.get<std::string>("time");
            time.replace(8, 1, "T");
            ptime t = from_iso_string(time);
            double p_val = array_element.second.get<double>("P");

            bsoncxx::builder::stream::document document{};
            auto builder = bsoncxx::builder::stream::document{};
            bsoncxx::document::value doc_value = builder
                << "time" << to_iso_extended_string(t)
```

```
                << "p" << p_val
                << bsoncxx::builder::stream::finalize;
            auto collection = conn["Terra"]["sun" + direction];
            collection.insert_one(doc_value.view());
        }
    }
    return true;
}

void fillData(boost::property_tree::ptree &pt, std::string time, mongocxx::client& conn)
{
    bsoncxx::builder::stream::document doc{};
    time = time.substr(0, 16);
    doc << "time" << time;

    for (auto & array_element : pt) {

        std::string elementId = array_element.second.get<std::string>("elementId");
        double value = array_element.second.get<double>("value");
        doc << elementId << array_element.second.get<double>("value");
    }


    auto collection = conn["Terra"]["weather"];
    collection.insert_one(doc.view());
}

void fillDoc(boost::property_tree::ptree &pt, std::string time, mongocxx::client& conn)
{
    if(pt.empty())
    {
        return;
    }
    else
    {
        for (boost::property_tree::ptree::iterator pos = pt.begin(); pos != pt.end();)
        {
            if(pos->first == "referenceTime")
            {
                time = pos->second.data();
            }
            else if(pos->first == "observations")
            {
                fillData(pos->second, time, conn);
            }
            fillDoc(pos->second, time, conn);
            ++pos;
        }
    }
}
```

```
bool fillWeather(mongocxx::client& conn)
{
    std::ifstream ist(weatherPath);
    ptree pt;
    read_json(ist, pt);
    bsoncxx::builder::stream::document doc{};
    std::vector<std::thread*> threads;

    fillDoc(pt, "", conn);

    return true;
}


bool fillPowerPrices(mongocxx::client& conn)
{
    const int dateIndex = 0;
    const int hourIndex = 1;
    const int osloIndex = 10;

    for(int year = 2013; year <= 2020; year++)
    {
        std::cout << "Year: " << year << "\n";
        std::ifstream file;
        file.open(powerPath + std::to_string(year) + powerExt);
        auto table = readCSV(file);
        for(int i = 3; i < table.size(); i++)
        {
            std::string date = table[i][dateIndex];
            std::string hour = table[i][hourIndex];
            std::string price = (table[i][osloIndex]);
            boost::replace_all(price, ",", ".");

            if(price.empty())
            {
                continue;
            }
            double priceNum = std::stod(price);
            std::string time = date.substr(6, 4) + "-" + date.substr(3, 2) + "-" +
                date.substr(0, 2) + "T" + hour.substr(0, 2) + ":00:00";

            bsoncxx::builder::stream::document document{};
            auto builder = bsoncxx::builder::stream::document{};
            bsoncxx::document::value doc_value = builder
                    << "time" << time
                    << "price" << priceNum
                    << bsoncxx::builder::stream::finalize;
                auto collection = conn["Terra"]["powerPrices"];
                collection.insert_one(doc_value.view());
```

```
        }
    }


    return true;
}
```

B.3 Historical Hourly Avg. Power Price

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn import metrics
import datetime as dt
import pymongo
from mongoConfigFile import *

hourPrices = [0] * 24
hourPoints = [0] * 24
hours = list(range(0, 24))

clientUrl =
    ("mongodb+srv://{0}:{1}@terrasolaris-robmp.azure.mongodb.net/test?retryWrites=true&w=majority".
     format(mongoUsername, mongoPassword))
client = pymongo.MongoClient(clientUrl)
db = client["Terra"]

def main():
   fillData()
   drawPower()

def fillData():
   print("Fyller priser..")
   collection = db["powerPrices"]
   #cursor = collection.find({"time": "2005-01-01T00:00:00"})
   cursor = collection.find({})
   for document in cursor:

      date = pd.to_datetime(document["time"])
      hourPrices[int(date.hour)] += document["price"]
      hourPoints[int(date.hour)] += 1

def drawPower():
   print("drawing")
   X = []
```

```
   for i in range(0, len(hours)):
      X.append(hourPrices[i] / hourPoints[i])

   y = hours

   objects = hours
   y_pos = np.arange(len(hours))
   performance = X

   plt.bar(y_pos, performance, align='center', alpha=0.5)
   plt.xticks(y_pos, objects)
   plt.ylabel('NOK per MWh')
   plt.xlabel('Hour of day')
   plt.title('Average hourly power price 2013-2020')

   plt.show()

main()
```

B.4 Machine Learning

Solar Power Production

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn import metrics
import datetime as dt
import pymongo
from mongoConfigFile import *

power = []
dates = []
weather = {}
data = {'Year': [], 'Month': [], 'Day': [], 'Hour': [], 'Temp': [],
'CloudAreaFraction': [], 'CloudBaseHeight': [], 'WeatherType': [], 'Power': []}
clientUrl =
    ("mongodb+srv://{0}:{1}@terrasolaris-robmp.azure.mongodb.net/test?retryWrites=true&w=majority".
     format(mongoUsername, mongoPassword))
client = pymongo.MongoClient(clientUrl)
db = client["Terra"]

def main():
   fillData()
   fillWeather()
```

```
    dataset = fillPanda()
    betterPredictWithWeather(dataset)

def fillData():
    print("Fyller sol..")
    collection = db["sunsouth"]
    cursor = collection.find({})
    for document in cursor:
        dates.append(document["time"])
        power.append(document["p"])

def fillWeather():
    print("Filling weather..")
    collection = db['weather']
    cursor = collection.find({})
    for document in cursor:
        arr = []
        arr.append(document.get('air_temperature'))
        arr.append(document.get('cloud_area_fraction'))
        arr.append(document.get('cloud_base_height'))
        arr.append(document.get('weather_type'))
        weather[document['time'] + ':00'] = arr


def fillPanda():
    print('Fyller Panda..')
    db = client["Terra"]
    collection = db["weather"]

    for date in dates:
        l = weather.get(date)
        date = pd.to_datetime(date)
        data['Year'].append(int(date.year))
        data['Month'].append(int(date.month))
        data['Day'].append(int(date.day))
        data['Hour'].append(int(date.hour))
        default = None
        data['Temp'].append(l[0] if l is not None and 0 < len(l) else default)
        data['CloudAreaFraction'].append(l[1] if l is not None and 1 < len(l) else default)
        data['CloudBaseHeight'].append(l[2] if l is not None and 2 < len(l) else default)
        data['WeatherType'].append(l[3] if l is not None and 3 < len(l) else default)

    data['Power'] = power
    df = pd.DataFrame(data=data)
    df.fillna(df.mean(), inplace=True)
    df = df.reset_index()
    return df

def betterPredictWithWeather(dataset):
```

```
    X = dataset[['Year', 'Month', 'Day', 'Hour', 'Temp', 'CloudAreaFraction',
        'CloudBaseHeight', 'WeatherType']].values
    y = dataset['Power'].values

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.01, shuffle=False)
    regressor = DecisionTreeRegressor(random_state = 0)
    regressor.fit(X_train, y_train)
    y_pred = regressor.predict(X_test)

    print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))

    X_dates = []
    for date in X_test:
        X_dates.append(dt.datetime(int(date[0]), int(date[1]), int(date[2]), int(date[3])))

    df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred}, index = X_dates)

    df1 = df.head(100)
    ax = df1.plot(kind='bar',figsize=(200,100))
    plt.grid(which='major', linestyle='-', linewidth='0.5', color='green')
    plt.grid(which='minor', linestyle=':', linewidth='0.5', color='black')

    x_labels = df1.index.strftime('%d-%b-%y %H:00')
    ax.set_xticklabels(x_labels)
    ax.set_xlabel('Date')
    ax.set_ylabel('Wh')
    plt.show()


main()
```

Machine Learning, Power Price Prediction

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn import metrics
import datetime as dt
import pymongo
from mongoConfigFile import *

powerPrices = []
dates = []
data = {'Year': [], 'Month': [], 'Day': [], 'Hour': [], 'PowerPrice': []}
clientUrl =
    ("mongodb+srv://{0}:{1}@terrasolaris-robmp.azure.mongodb.net/test?retryWrites=true&w=majority".
```

```
      format(mongoUsername, mongoPassword))
client = pymongo.MongoClient(clientUrl)
db = client["Terra"]

def main():
   fillData()
   dataset = fillPanda()
   betterPredict(dataset)

def fillData():
   print("Fyller priser..")
   collection = db["powerPrices"]
   #cursor = collection.find({"time": "2005-01-01T00:00:00"})
   cursor = collection.find({})
   for document in cursor:
      dates.append(document["time"])
      powerPrices.append(document["price"])

def fillPanda():
   print('Fyller Panda..')

   for date in dates:
      date = pd.to_datetime(date)
      data['Year'].append(int(date.year))
      data['Month'].append(int(date.month))
      data['Day'].append(int(date.day))
      data['Hour'].append(int(date.hour))

   data['PowerPrice'] = powerPrices
   df = pd.DataFrame(data=data)
   df.fillna(value=pd.np.nan, inplace=True)
   df = df.reset_index()
   return df


def betterPredict(dataset):
   X = dataset[['Year', 'Month', 'Day', 'Hour']].values
   y = dataset['PowerPrice'].values

   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.01,
       shuffle=False)#, random_state=0)

   regressor = DecisionTreeRegressor(random_state = 0)
   regressor.fit(X_train, y_train)
   y_pred = regressor.predict(X_test)

   X_dates = []
   for date in X_test:
      X_dates.append(dt.datetime(date[0], date[1], date[2], date[3]))
```

```
    df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred}, index = X_dates)

    df1 = df.head(50)
    ax = df1.plot(kind='bar',figsize=(200,100))
    plt.grid(which='major', linestyle='-', linewidth='0.5', color='green')
    plt.grid(which='minor', linestyle=':', linewidth='0.5', color='black')

    x_labels = df1.index.strftime('%d-%b-%y %H:00')
    ax.set_xticklabels(x_labels)
    ax.set_xlabel('Date')
    ax.set_ylabel('NOK/MWh')
    plt.show()

    print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))

main()
```

B.5 Web-Application

script.js

```
//Every room area is built with x,y coordinates.  Always starting at top left corner, then
    going clockwise
function Plan(width,height) {
  this.bedroom1={color:"#C4CDE0",
  name:"Soverom 1",
  area:
  width*3/5+","+0+" "
  +width+","+0+" "
  +width+","+height*1/6+" "
  +width*3/5+","+height*1/6,
  x:width*3.5/5,y:height*1/6/2};

  this.bedroom2={color:"#CDE0C4",
  name:"Soverom 2",
  area:
  width*3/5+","+height*1/6+" "
  +width+","+height*1/6+" "
  +width+","+height*2/6+" "
  +width*3/5+","+height*2/6,
  x:width*3.5/5,y:height*1/4};

  this.bedroom3={color:"#CDE0C4",
  name:"Soverom 3",
  area:
  0+","+height*5/6+" "
  +width*2/6+","+height*5/6+" "
```

```
+width*2/6+","+height+" "
+0+","+height,
x:width*1/10,y:height*11/12};

this.hallway={color:"#A09CB3",
name:"Gang",
area:
0+","+0+" "
+width*3/5+","+0+" "
+width*3/5+","+height*3/6+" "
+width*2/5+","+height*3/6+" "
+width*2/5+","+height*1/12+" "
+0+","+height*1/12,
x:width*1/5,y:height*0.6/12};

this.office={color:"#C4CDE0",
name:"Kontor",
area:
0+","+height*1/12+" "
+width*2/5+","+height*1/12+" "
+width*2/5+","+height*1.5/6+" "
+0+","+height*1.5/6,
x:width*1/6,y:height*1/6};

this.livingroomAndKitchen={color:"#E0C4CD",
name:"Stue og kjokken",
area:
0+","+height*2.5/6+" "
+width*2/5+","+height*2.5/6+" "
+width*2/5+","+height*3/6+" "
+width+","+height*3/6+" "
+width+","+height+" "
+width*2/6+","+height+" "
+width*2/6+","+height*5/6+" "
+0+","+height*5/6+" ",
x:width*1/2,y:height*3/4};

this.bathroom1={color:"#C4DBE0",
name:"Bad 1",
area:
width*3/5+","+height*2/6+" "
+width+","+height*2/6+" "
+width+","+height*3/6+" "
+width*3/5+","+height*3/6,
x:width*3.5/5,y:height*4.2/10};

this.bathroom2={color:"#C4DBE0",
name:"Bad 2",
area:
0+","+height*1.5/6+" "
```

```
  +width*2/5+","+height*1.5/6+" "
  +width*2/5+","+height*2.5/6+" "
  +0+","+height*2.5/6,
  x:width*1/6,y:height*2/6};
};

function applyTempIndexEventListeners(){
  //temperatureValueArray contains temperatures matching the index from Plan obj.
  let container = document.getElementById('svgContainer');
  let svg = document.createElementNS("http://www.w3.org/2000/svg","svg");
  let width = document.body.clientWidth;
  let height = width*2;
  let romPlan = new Plan(width,height);
  let keys = Object.keys(romPlan);
  for (var i = 0; i < keys.length; i++) {
    let tempGroup = document.createElementNS("http://www.w3.org/2000/svg","g");
    let roomName = romPlan[keys[i]].name
    tempGroup.setAttribute("data-url",roomName);
    tempGroup.addEventListener("click",function(){
      let roomName = this.getAttribute("data-url");
      //console.log(roomName);
      window.location.href = "/temperatur/endre?rom="+roomName.toLowerCase();
    });
    let tempPolygon = document.createElementNS("http://www.w3.org/2000/svg","polygon");
    let tempTextRoom = document.createElementNS("http://www.w3.org/2000/svg","text");
    tempTextRoom.innerHTML = roomName;
    tempTextRoom.setAttribute("x",romPlan[keys[i]].x);
    tempTextRoom.setAttribute("y",romPlan[keys[i]].y);
    let tempTextTemperature =
        document.createElementNS("http://www.w3.org/2000/svg","text");
    tempTextTemperature.innerHTML = temperatureValueArray[i]+"C";
    tempTextTemperature.setAttribute("x",romPlan[keys[i]].x);
    tempTextTemperature.setAttribute("y",romPlan[keys[i]].y+20);
    tempPolygon.setAttribute("points",romPlan[keys[i]].area);
    tempPolygon.setAttribute("stroke","black");
    tempPolygon.setAttribute("fill",romPlan[keys[i]].color);
    tempGroup.appendChild(tempPolygon);
    tempGroup.appendChild(tempTextRoom);
    tempGroup.appendChild(tempTextTemperature);
    svg.appendChild(tempGroup);
  }

  svg.setAttribute("width",width);
  svg.setAttribute("height",height);


  container.appendChild(svg);
}

function findRoomId(roomName){
```

```
  let romPlan = new Plan(0,0);
  let keys = Object.keys(romPlan);
  for (var i = 0; i < keys.length; i++) {
    if (roomName==romPlan[keys[i]].name) {
      return i+1;
    }
  }
}

function applyTempChangeEventListeners(){
  //Smaller variable = quicker temperature update
  var responsiveness = 50;

  var currentAngle = 0;
  var tempChange = 0;

  var roomId = findRoomId(document.getElementById("romDiv").innerText);
  setInitalTemperature(roomId);

  var temperatureWheel = document.getElementById('temperatureWheel');
  var region = new ZingTouch.Region(temperatureWheel);
  region.bind(temperatureWheel, 'rotate', function(e){
    tempChange += e.detail.distanceFromLast;
    currentAngle += e.detail.distanceFromLast;
    temperatureWheel.style.transform = 'rotate(' + currentAngle + 'deg)';
    let temperatureDerivedFromAngle = Math.round(tempChange/responsiveness);
    if(temperatureDerivedFromAngle != 0){
      tempChange = 0;
      updateRoomTemperature(temperatureDerivedFromAngle,roomId);
    }
  });
}

function setInitalTemperature(id){
  var roomTemperatureDiv = document.getElementById('degrees');
  var oReq = new XMLHttpRequest();
  oReq.addEventListener("load", function(){
    let data = JSON.parse(this.response);
    roomTemperatureDiv.innerText = data.temperatureValue;
  });
  oReq.open("POST", "/temperatur/getSpecificTemperature");
  oReq.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
  oReq.send("temperatureId="+id);
}

function updateRoomTemperature(degrees,id){
  let roomTemperatureDiv = document.getElementById('degrees');
  let oldTemperature = parseInt(roomTemperatureDiv.innerText);
  let newTemperature = oldTemperature+degrees;
  //User limit on temperature control
```

```
  if(newTemperature<=30 && newTemperature>=15){
    roomTemperatureDiv.innerText = newTemperature;
    var oReq = new XMLHttpRequest();
    oReq.addEventListener("load", function(){
    });
    oReq.open("POST", "/temperatur/updateSpecificTemperature");
    oReq.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    oReq.send("temperatureId="+id+"&temperatureValue="+newTemperature);
  }
}
```

nav.php

```
<?php
$action = $this->getAction();
?>
<nav class="navbar navbar-light bg-light">
  <img style='width: 120px' src="/img/logo2.png">
  <a href="/temperatur"><button class="btn-lg btn-primary"
      type="button">Tilbake</button></a>
  <a href="/"><button class="btn-lg btn-primary" type="button">Meny</button></a>
</nav>
```

header.php

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>TSB App</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="/main.css" type="text/css" media="all" rel="stylesheet">
  <link href="/lib/bootstrap/bootstrap.min.css" type="text/css" media="all"
      rel="stylesheet">
  <script
      src='https://cdnjs.cloudflare.com/ajax/libs/zingtouch/1.0.6/zingtouch.min.js'></script>
</head>
<body class='bg-light'>
<script src='/script.js'></script>
```

footer.php

```
</body>
```

```
</html>
```

Temperatur index.php

```
<?php include VIEW.'header.php';?>
<?php include VIEW.'nav.php';?>
<div class='container-fluid p-0'>
  <h1 class="text-center">
    Velg rom
  </h1>
  <div id='svgContainer' class='media'>
  </div>
</div>
<script>
  let temperatureValueArray = [
  <?php
  foreach ($this->viewData as $row) {
      echo $row['temperatureValue'].",";
  }
  ?>
  ];
  applyTempIndexEventListeners();
</script>
<?php include VIEW.'footer.php';?>
```

endre.php

```
<?php include VIEW.'header.php';?>
<?php include VIEW.'nav.php';?>
<div class='container-fluid'>
  <h1 class="text-center">
    <div class='d-inline-block' id='romDiv'>
      <?php
        echo ucfirst("$this->viewData");
      ?>

    </div>
    <br>
    <div class='d-inline-block'> temperatur:</div>
    <div class='d-inline-block' id='degrees'></div>
    <div class='d-inline-block' id='celcius'>C</div>
  </h1>
  <div id='temperatureWheelContainer' class='media ml-5 mr-5'>
    <img class='img-fluid' data-angle='0' id='temperatureWheel' src="/img/wheel.png">
  </div>
</div>
```

```
<script>
  applyTempChangeEventListeners();
</script>
<?php include VIEW.'footer.php';?>
```

index.php

```
<?php include VIEW.'header.php';?>
<div style='text-align: center;'>
<img style='width: 200px;' src="/img/logo2.png">

<h1>Energi kontroll</h1>
<h3>Leilighet 5 D</h3>
</div>
<div class="indexDiv">
   <a href="/temperatur"><button class="btn-lg btn-primary" type="button">Endre temperatur
       i leilighet</button></a>

   </div>
   <div class="indexDiv">
   <a href="/"><button class="btn-lg btn-primary" type="button">Oversikt over
       stromforbruk</button></a>

   </div>
    <div class="indexDiv">
    <a href="/"><button class="btn-lg btn-primary" type="button">Endre lade-innstillinger
        for el-bil</button></a>
    </div>

<?php include VIEW.'footer.php';?>
```

TemperaturModel.php

```
<?php
class TemperaturModel extends Model{

  public function getAllTemperatures(){
    $data = $this->query("SELECT temperatureValue FROM temperatureTable")->fetchAll();
    return $data;
  }

  public function getSpecificTemperature($id){
    $data = $this->query("SELECT temperatureValue FROM temperatureTable WHERE
        temperatureId = $id")->fetch();
    return $data;
  }
```

```
  public function updateSpecificTemperature($temperature,$id){
    $data = $this->query("UPDATE temperatureTable SET temperatureValue = $temperature
        WHERE temperatureId = $id")->fetch();
    return $data;
  }
}
?>
```

TemperaturController.php

```
<?php
class TemperaturController extends Controller{

  public function updateSpecificTemperature(){
    $this->model('TemperaturModel');
    $temperatureId = $_POST['temperatureId'];
    $temperatureValue = $_POST['temperatureValue'];
    $this->model->updateSpecificTemperature($temperatureValue,$temperatureId);
  }

  public function getSpecificTemperature(){
    $this->model('TemperaturModel');
    $temperatureId = $_POST['temperatureId'];
    $data = $this->model->getSpecificTemperature($temperatureId);
    echo json_encode($data);
  }

  public function endre(){
    if(count($_GET) > 0){
      $this->view('temperatur'.'/'.'endre.php',$_GET['rom']);
      $this->view->render();
    } else {
      header("/");
    }
  }
  public function index(){
    $this->model('TemperaturModel');
    $temperatures = $this->model->getAllTemperatures();
    $this->view('temperatur'.'/'.'index.php',$temperatures);
    $this->view->render();

  }
}
?>
```

HomeController.php

```
<?php
class HomeController extends Controller{
  public function index(){

    $this->view('home'.'/'.'index.php');
    $this->view->render();
  }
}
?>
```

Application.php

```
<?php
class Application{
  protected $controller = 'HomeController';
  protected $action = 'index';

  public function __construct(){
    $this->prepareURL();
    $this->controller = ucfirst($this->controller);
    if(file_exists(CONTROLLER.$this->controller.'.php')){
      $this->controller = new $this->controller;
      if(method_exists($this->controller,$this->action)){
        call_user_func_array([$this->controller,$this->action],[]);
      }
    }
  }
  protected function prepareURL(){
    $request = trim($_SERVER['REQUEST_URI'],'/');
    if(!empty($request)){
      $token = explode('?', $request);
      $urlArray = explode('/',$token[0]);
      if(isset($urlArray[0])){
        $this->controller = $urlArray[0].'Controller';
      }
      if(isset($urlArray[1])){
        $this->action = $urlArray[1];
      }
    }
  }
}
?>
```

Model.php

```
<?php
class Model extends PDO{
  private $dsm = 'mysql:dbname=tsb;host=localhost;charset=UTF8';
  private $user = 'tsb';
  private $pw = '';
  public function __construct(){
    parent::__construct($this->dsm,$this->user,$this->pw);
    }
}
?>
```

View.php

```
<?php
class View{

  protected $viewData;
  protected $viewFile;

  public function __construct($viewFile,$viewData){
    $this->viewData = $viewData;
    $this->viewFile = $viewFile;
  }
  public function getAction(){
    return (explode('/',$this->viewFile)[1]);
  }
  public function render(){
    if(file_exists(VIEW.$this->viewFile)){
      include VIEW.$this->viewFile;
    }
  }
}
```

Controller.php

```
<?php
abstract class Controller{
  protected $model;
  protected $view;
  public function model($modelName){
    $this->model = new $modelName;
  }
  public function view($viewName,$data=[]){
```

```
    $this->view = new View($viewName,$data);
  }
}
```