

Sami Näres

IoT as a Smart Home Implementation

Automation of Air Humidifier

Helsinki Metropolia University of Applied Sciences

Bachelor of Engineering

Media Engineering

Thesis

21 April 2016

Author(s) Title Number of Pages Date	Sami Näres IoT as a Smart Home Implementation: Automation of Air Humidifier 33 pages + 1 appendix 21 April 2016
Degree	Bachelor of Engineering
Degree Programme	Media Engineering
Instructor	Kari Salo, Principal Lecturer
<p>The purpose of the thesis is to explain IoT as a technology and provide an overview of its current situation and to the application it can be used for. In practice, the benefits of IoT were demonstrated by embedding IoT technology into a house appliance. The goal for the project was to provide optimization and improvement to a regular household air humidifier.</p> <p>The humidifier was controlled with a Wi-Fi switch plug, which was accessed with an SSH connection from an Apache server. Thingsee was used to measure air humidity and provide data for the server. The project also included a website for the user to keep track of the system's activity. Finally, the system was tested and compared against regular use of the humidifier to draw a conclusion of its benefits.</p> <p>Test results showed that the system was able to prolong the humidifier's lifetime by 77% while maintaining an average air humidity of 34%.</p> <p>The project was carried out successfully. The test results showed that the IoT implementation was able to control air humidity levels as intended and prolong the life time of the air humidifier between refills compared to regular use. Further studies should research the possibility of building an automated water refill system for the humidifier.</p>	
Keywords	IoT, air humidity, smart home, automation, Thingsee, PHP

Tekijä Otsikko	Sami Näres IoT:n käyttö älykotisovelluksessa
Sivumäärä Päivämäärä	32 sivua + 1 liite 6.4.2016
Tutkinto	Bachelor of Engineering
Koulutusohjelma	Media Engineering
Ohjaaja	Yliopettaja Kari Salo
<p>Insinööriyön tarkoituksena oli perehtyä esineiden internetiin (Internet of Things, IoT) teknologiana, sen nykytilaan ja sen mahdollistamiin sovelluksiin. IoT:n hyötyä demonstroitettiin liitettämällä teknologiaa kodinkoneeseen. Insinööriyön tavoite oli optimoida ja parantaa tavallisen ilmastuttimen toimintaa.</p> <p>Ilmastutinta hallittiin pistorasian kautta, joka oli liitetty langattomaan verkkoon. Pistorasiaan muodostettiin yhteys Apache-palvelimelta Secure Shell (SSH) -protokollaa käyttämällä. Monitoimilaite Thingseeä käytettiin ilmastusteuden mittaamiseen ja tiedon välittämiseen palvelimelle. Osana projektia rakennettiin myös verkkosivu, jolta käyttäjä voi tarkkailla järjestelmän toimintaa. Lopuksi järjestelmä testattiin ja tuloksia verrattiin keskenään, jotta voitiin saada johtopäätös sen tuomista eduista.</p> <p>Koetulokset osoittivat, että järjestelmä pidentä ilmastuttimen käyttöikää vesitankin täyttöjen välillä 77 prosenttia samalla ylläpitäen ilmastusteuden keskiarvon 34 prosentissa.</p> <p>Projekti vietiin loppuun onnistuneesti. Koetulokset todistivat, että IoT-sovellus valvoi ilmastusteutta sille annettujen parametrien mukaisesti ja pidentä ilmastuttimen käyttöikää vesitankin täyttöjen välillä. Jatkoa ajatellen voitaisiin tutkia automaattisen vesipumppujärjestelmän rakentamista ja käyttöönottoa.</p>	
Avainsanat	IoT, ilmastusteus, älykoti, automaatio, Thingsee, PHP

Contents

Abstract

Abbreviations

1	Introduction	1
2	Internet of Things	2
2.1	History	3
2.2	Market	4
2.3	Applications	6
3	The Final Year Project	7
4	Hardware	8
4.1	Thingsee	8
4.2	Kankun Wi-Fi Switch Plug	10
4.3	Air Humidifier	11
5	Software	13
5.1	XAMPP	13
5.2	Thingsee Creator	13
5.2.1	REST Calls	14
5.3	Google Charts API	15
5.3.1	Overview	15
5.3.2	Customization	16
5.4	Facebook Application	17
5.4.1	Graph API	17
5.4.2	PHP SDK	17
5.5	HTTP	18
5.6	Diversity of HTTP Communication	19
5.7	MQTT	20
6	Implementation	21
6.1	Thingsee Configuration	21
6.2	Back-end	22
6.2.1	Database	23
6.2.2	Communication with Kankun	23

6.2.3	Water-level Counter	24
6.2.4	Facebook Notification	24
6.3	Web App Monitor	25
7	Testing	27
8	Discussion and Evaluation	28
8.1	Results	28
8.2	Comparison	29
8.3	Evaluation	29
8.3.1	The Task of The Water Humidifier	29
8.3.2	Test Area	30
8.3.3	Thingsee	30
8.3.4	Security	31
9	Conclusion	32
	References	33

Appendix 1. Test Measurements

Abbreviations

IoT	Internet of Things
M2M	Machine to Machine
NAT	Network Address Translation
API	Application Programming Interface
PHP	Hypertext Pre-processor
HTML	Hypertext Mark-up Language
REST	Representational State Transfer
SQL	Structured Query Language
JSON	JavaScript Object Notation
SSH	Secure Shell
HTTP	Hypertext Transfer Protocol
MQTT	Message Queue Telemetry Transport

1 Introduction

Computers and the internet have made their way from large halls and government secluded buildings first to common people households and during the last 10 years even to people's pockets. This level of access to technology lets people share and interact whatever, whenever and almost where-ever they are. The next stage of this technological advancement is called the Internet of things (IoT) and it can be seen starting to appear everywhere in infrastructure from households and corporate offices to even cars and public areas.

The purpose of this thesis is to explain what IoT is, how it can be used for and what possibilities it offers. For the practical part of the final year project IoT was used to optimize and automate a consumer market air humidifier in a student apartment. Measurement of air humidity was done with a commercial multi-tool measurement device called Thingsee, which provided the data for the PC that controlled a Wi-Fi switch plug between the humidifier and a wall outlet. The thesis will explain the different technologies used and document the steps of the setup process.

The thesis was carried out for Helsinki Metropolia University of Applied Sciences in Espoo Finland to provide an example of a case for developers of how the Thingsee device works and how it can be used for this type of projects. IoT devices are becoming more common electronics in stores and this opens up the possibility for everyone to build their own network of devices with only the imagination as the limit.

2 Internet of Things

The Internet of Things is believed to have long lasting effects in both technology and modern society. In modern information society, IoT can be seen as a global infrastructure that enables more advanced services by connecting physical and virtual devices and things to currently existing and even upcoming information and communication technologies.

IoT takes advantage of identification, data capture, processing and communication capabilities of modern technology to allow regular machines to provide new data sources to applications, which in turn can offer more advanced services. In terms of ICT technologies, IoT adds Any Thing communication to Any Time and Any Place as illustrated by figure 1.

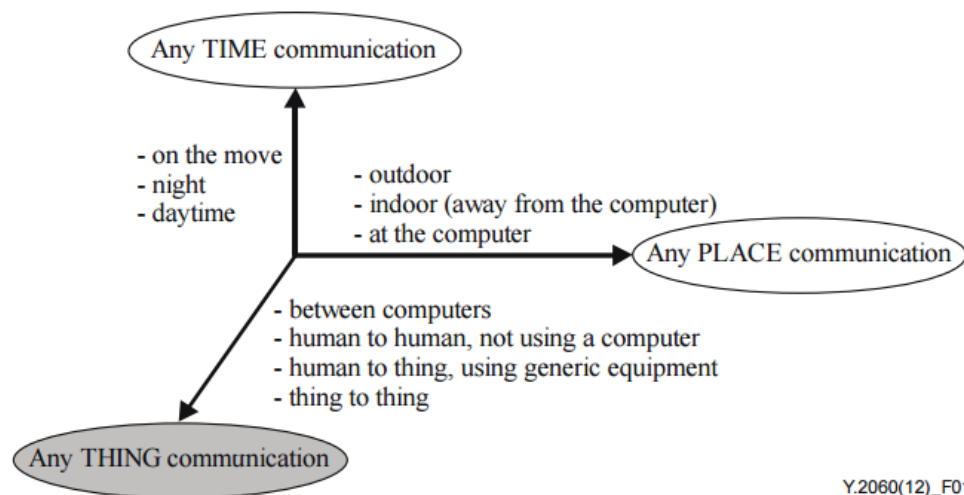


Figure 1. IoT compared to other ICT technologies. Copied from International Telecommunication Union (2012).

IoT devices are both physical objects and things part of the virtual world, which can be identified and connected to networks. The information they contain can be static and dynamic. (International Telecommunication Union 2012.)

Kevin Ashton was the first to use the term IoT in a speech in 1999, in which he said that computers and the Internet are almost fully relying on people to provide data. A rough estimate is that around 50 petabytes of information found from the Internet has been either written, scanned or recorded by humans. People are literally routers between the

real and virtual world. The problem with people is their limited time, attention and accuracy by which they record and transfer data to computers. Computers are now relying more on people's ideas of things instead of the things themselves. With the information from things, computers could count everything and greatly improve the way the infrastructure works. In such scenario people could just take care of maintenance and replace machines when needed. (Ashton 2009.)

2.1 History

Even though the term IoT has started to trend in public during the last five years, connecting things to the Internet is not a new phenomenon. Possibly the first application was the Trojan room coffee pot that was invented shortly after internet was born in 1989. The Trojan room coffee pot was a webcam providing live footage of an office coffee pot to the workers at University of Cambridge. The workers could check from a computer screen whether or not the pot had any coffee in it. (Cambridge 201.)

In 2003 there were approximately 0.08 devices connected to internet per each person on the planet. With introduction of smartphones the number of devices started to grow rapidly. In 2010 the relative number had reached 1.84 with 12.5 billion devices online. Some argue IoT was "born" around 2008-2009. (Evans 2011.)

The year 2011 was big for IoT because earlier one of the biggest limitations for the technology was the small address capacity of IPv4 and NAT, which enabled users to share the same public IP addresses. This was problematic because the same sensors acting as end-points could have been used by different stakeholders. (IoT6, 2014.) IPv6 was answer to this problem with its scalable address scheme, which increased the address length from 32 to 128 bits, thus creating 2^{96} new IP addresses (Hanumanthappa 2008). Gartner also noted the increasing presence of IoT in 2011 and for the first time the term appeared on Gartner's hype cycle of emerging technologies graph (as seen in figure 2). Three years later in 2014 the term was at the top in the "Peak of Inflated expectations". (Gartner 2014.)

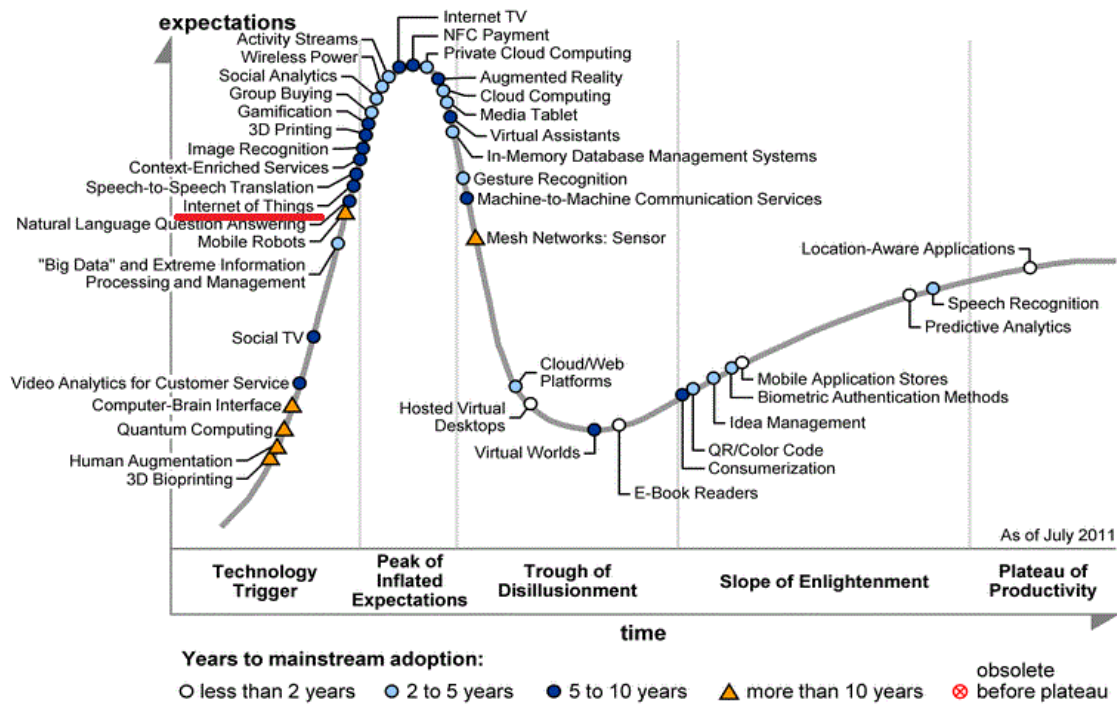


Figure 2. Gartner's hype cycle for emerging technologies 2011. Copied from Gold-Group (2011).

Another indicator of the IoT trend can be seen in Google trends, which visualizes the search interest of the term with numbers relative to the highest point in the chart as seen in figure 3.

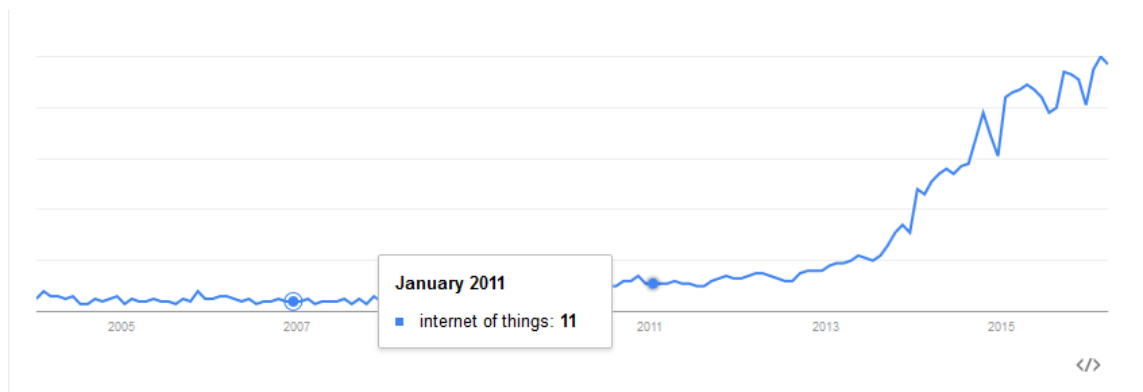


Figure 3. Search Interest in IoT visualized in Google Trends. Copied from Google (2016).

2.2 Market

IoT has already found its way to a variety of different markets, and predictions indicate a strong growth for the next 15 years. Applications are aimed for consumers and companies alike although a certain trend can be seen in how these target groups are divided among the markets (see Figure 4).

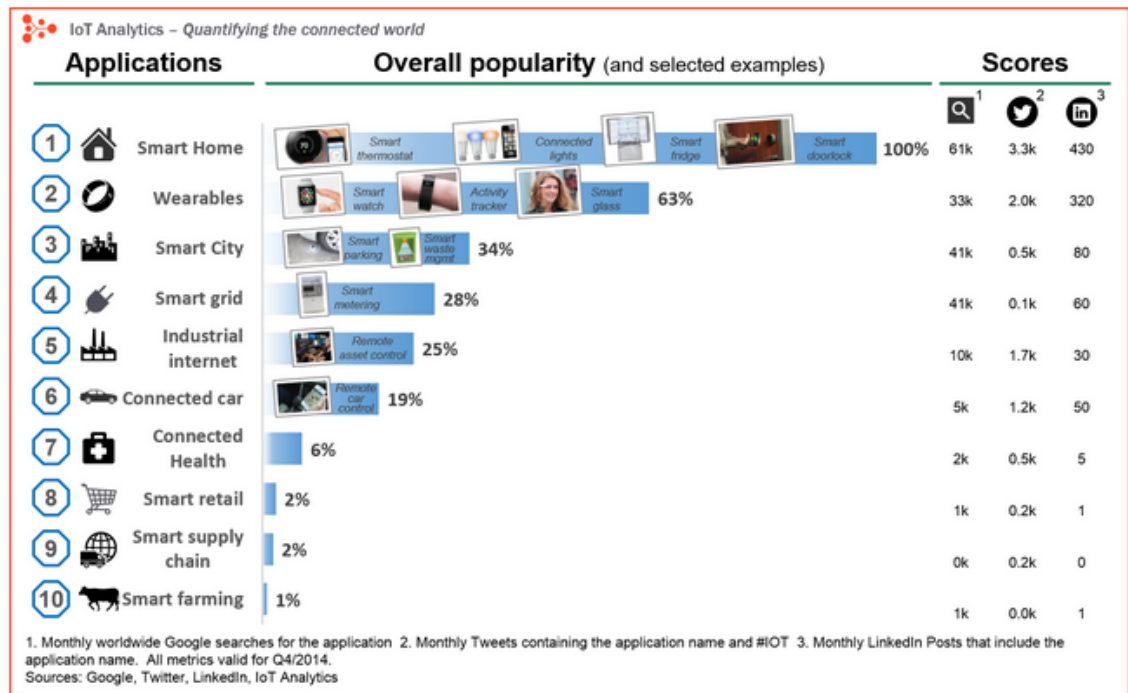


Figure 4. IoT application ranking. Copied from IoT-Analytics (2015).

The most popular market is smart homes. The term has more than 60,000 search hits monthly and start-up companies have received over 2.5 billion dollars in total from funding. Examples of these companies are Philips, Belkin, Nest and AlertMe. Home automation solutions include lightbulbs, home appliances and security devices. (IoT-Analytics 2015)

The opportunities IoT offers for businesses are huge as the demand and interest are both clearly visible. Companies should now shift their focus to use cases that would allow organizations to get most out of IoT by adapting new business processes and improving effectiveness of operations. (PTC 2015.)

Surprisingly majority of organizations have not started to create value from IoT with the biggest challenges being undeveloped business cases and unclear monetization plans. (PTC 2015.)

2.3 Applications

Industries are considering and identifying a huge amount of IoT-related applications, which can be divided into two categories. In first category the devices are connected, forming an infrastructure that is automated with M2M communication and aiming to simplify people's lives. In this category IoT can be seen playing the role of TCC&R (track, command and control). In households for example the room temperature, windows, lights and electrical devices etc. can all be controlled remotely from laptop and automated to get rid of manual processes people face daily in their lives.

Keeping track of companies' assets becomes easier, when machines and resources are constantly sending information to servers. Equipment malfunctions become less frequent when they are under real time surveillance and their condition is constantly observed. Another big topic has been smart homes, in which the air conditioning, lights, doors and appliances are controlled through smart phone applications. IoT is also believed to revolutionize healthcare. Devices can collect patient data, monitor vital signs and automatically adjust medication. (Freescale 2014.)

In the second category, the devices are data mines that monitor trends and behaviours in order to provide companies with marketing information and to create commerce. This category has raised most concern about the privacy of the users, and to what extent they want to share data about themselves is then used to categorize them. (Freescale 2014.)

3 The Final Year Project

The system presented in this final year project consisted of a home server, a Thingsee multi-tool measurement device, a Kankun Wi-Fi plug and a commercial air humidifier. The goal was to automate and optimize the humidifier's operating principle by monitoring air humidity levels inside a student apartment and running a server that manages how often the humidifier is turned on. The humidifier used in this project has an operating time of 16-20 hours between refills and this was the target time to beat to demonstrate the usefulness of the system. Cold Climate Housing Research Centre in Alaska recommends an air humidity of 30-40% (CCHRC 2014.) for houses located in cold climates. Test area was located in Espoo, Finland, which is also a cold climate area. This research chose a median of 35% as the target value to be maintained. The layout of the system can be seen in figure 5.

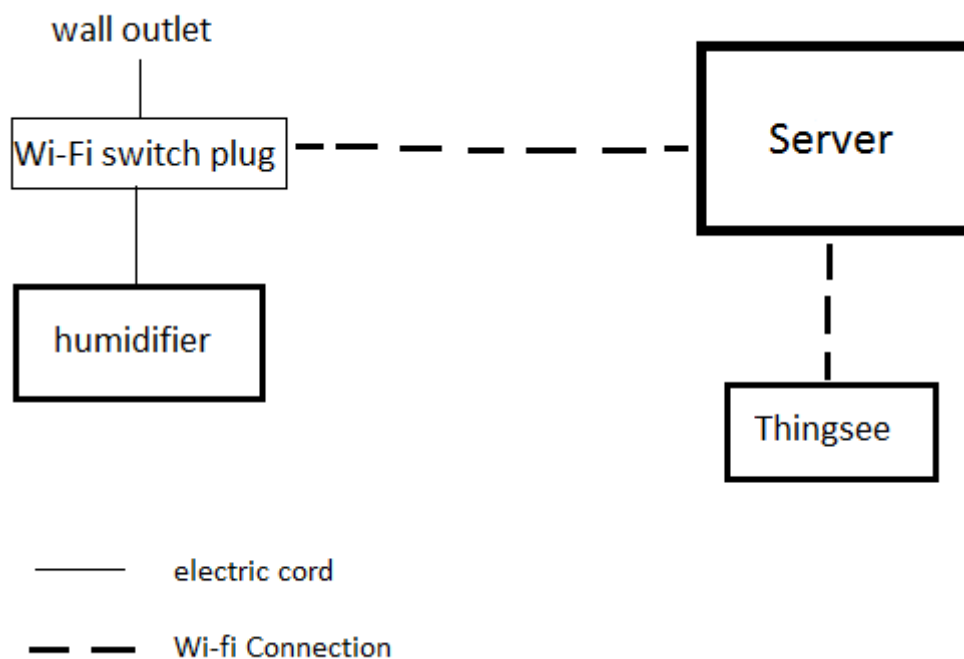


Figure 5. System Design

The system can be divided into the following segments:

- Hourly measurement and transfer of air humidity value to server
- Keeping track of water level in air humidifier by calculating water emitted per hour
- Turning Wi-Fi plug on and off depending on the water level and air humidity

- Sending a Facebook notification to user when water tank is running empty
- Providing user with detailed monitoring interface

4 Hardware

It should be noted that the hardware parts described in chapter 4 are not the only solutions to carry out this final year project, but the design placed some limitations which are explained individually.

4.1 Thingsee

Thingsee is a multi-purpose, open ecosystem IoT device capable of measuring its own location, speed and several variables from its surroundings using inbuilt sensors and taking advantage of GPS satellites. Thingsee's open ecosystem lets the user customize the applications as well as the device's firmware with their own API and rest calls. Haltian, the company behind Thingsee, states that in some use-case scenarios the device can be powered for a full year with one battery charge (Thingsee 2016). Another feature are the smart events and actions that can be triggered by sensor data from user configuration. Haltian released a commercial version of the device called Thingsee One in 2015 (seen in figure 6). The same device has been used in this project.



Figure 6. Thingsee One. Copied from MLove (2016).

To configure the device, the user is given a back-end platform hosted by Haltian that allows the user to create and manage dynamic state machines using a purpose API. The device is also capable of sending SMS alerts and push notifications to the user's phone, which requires a SIM card with an active 3G connection. The following list provides a more detailed technical specification of the hardware (Thingsee 2016):

Microcontroller Unit

- Ultra-low-power 32-bit MCU, ARM® based Cortex®-M3 with 512KB Flash, 80KB SRAM, 16KB EEPROM, AES- 128bit encryption hardware

Memory

- Micro SD memory card slot for data storage on device, support up to 128GB

Hardware User Interface

- Display: Standard 1.54" Monochrome Graphic OLED (128x64), white color
- Capacitive UI input via CapSense® MBR3108

Wireless Connectivity

- 2.5G GSM/GPRS Modem
- GSM 850/900/1800/1900 MHz
- GPRS Class 10, CS1-CS4 - up to 85.6kb/s
- WLAN IEEE 802.11 b/g
- Bluetooth LE 4.1

Size

- 110 x 67 x 19 mm

Sensors

- Ultra low-power 3-Axis accelerometer ST LIS2DH
- Humidity and temperature sensor ST HTS221
- Pressure sensor ST LPS25H
- iNEMO 9-axis inertial module (Gyroscope, Magnetometer)
- Ultra low-power Ambient light sensor MAX44009

4.2 Kankun Wi-Fi Switch Plug

Kankun is a Wi-Fi switch plug, that attaches between wall outlet and the electrical device the user wants to control. The units are not currently EU-compatible and require adapters for both the wall and the controlled device. The device can be seen in figure 7.



Figure 7. Kankun Wi-Fi switch plug unit KK-SP3. Copied from One2more (2016).

Most Wi-Fi switch plug manufacturers are aware of the security issues with their devices and force users to interact with their products with a mobile application. The application is paired with the device and is then the only way user can control the relay. In this sense Kankun “Small K” (KK-SP3) is not any different as it needs to be setup through the Smart Plug app that can be downloaded from the Apple app store.

However, Kankun is running a version of OpenWRT, a Linux environment that has SSH enabled by default. The connection requires a username and password, but these can be found online. With proper tools, the user can access and control KK-SP3 directly from

the command line and because PHP has SSH extensions, scripts can be used as well. (Burgener, 2015.)

4.3 Air Humidifier

Air humidifiers usually consist of a water tank, interface panel and a mechanism that turns the water into vapour and releases it into the air. The final year project used Wilfa HU-6W (seen in figure 8) that has the following specifications (Wilfa 2015):

- High capacity: cold steam 300 ml/h and warm steam 400 ml/h
- Adjustable steam emission level
- Ultrasonic
- Humidifies area up to 125 m²
- 6 litre water tank
- operating time with one tank 15-20 hours
- notification LED when the tank is empty



Figure 8. Wilfa HU-6W Ultrasonic air humidifier

This model was chosen based on the large capacity of the water tank to maximize the time it can run between refills and the area it can cover. The project was carried out in a

2-room 40 m² apartment, but the measurements were only taken in 10 m² living room where the humidifier was placed. For this setup to work, the humidifier needed to have a mechanical on/off switch, which is left in “on” position at all times. Digital switch would have had to be turned on again manually if the electrical current is cut from the wall outlet. Adjustable humidity emission level proved also to be useful to optimize humidifier’s running time between refills of the water tank.

5 Software

5.1 XAMPP

XAMPP is a free and open source cross-platform web server solution stack package, including the Apache HTTP Server, MySQL and support for PHP, Perl and Java. XAMPP is designed to help website developers and designers to test their products from any computer even when internet access is not available. Although XAMPP is designed for testing purposes, it can also be used to host websites online (XAMPP 2016). XAMPP is an acronym for:

- X (Cross-platform)
- Apache HTTP server
- MySQL
- PHP
- Perl
- Tomcat

5.2 Thingsee Creator

Haltian offers free backend service for Thingsee users called Thingsee Creator. The user creates a free account where he/she can register devices, create and manage “Purposes” and configure device settings. Purposes are rules created by the user for telling the device how often and which sensors to use (seen in figure 9). Purposes also specify where to send data and if SMS or push notification are used. Purpose has a starting state, which is the first stage of action the device takes. The user can configure, add and delete states depending on the complexity of a desired purpose. A purpose can have at least ten states and multiple tasks on the same state. The maximum amount of states or tasks is not defined but at least ten of each could be added. On the state tab users can define to which state to jump after the present one has been completed. The interface can be viewed as a user friendly algorithm builder for measuring and sending data. The device is capable of running up to three purposes simultaneously. The user can provide a clear description for all the purposes, states and tasks.

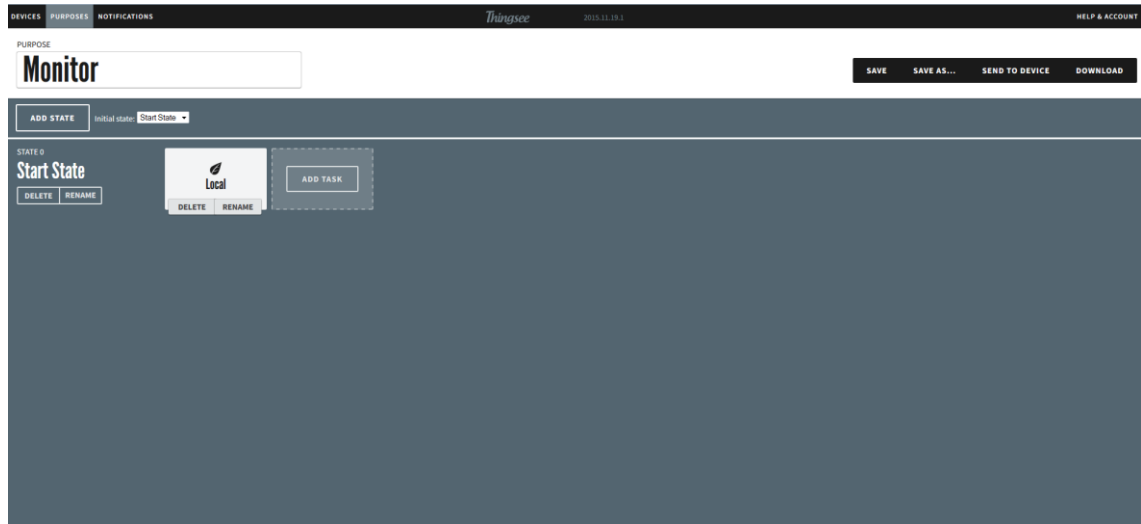


Figure 9. Dashboard view of the purpose management page in Thingsee Creator

Thingsee Creator can also be used as the monitor for the device where the user can see the values of the sensor measurements. Haltian also offers a mobile application for the same purpose. If the user decides to configure the tasks to send data to a custom URL, Haltian has included documentation for their Thingsee API for instructions on how to build a server that is capable of receiving that data in the backend created by the user.

5.2.1 REST calls

REST calls are made on command line using a curl command, and they are used to acquire information about the device. These calls can be sent straight to the device (assuming it is connected to internet) or to the Haltian backend (seen in figure 10).

```
[saminar@shell ~]$ curl -H "Content-Type:application/json" -X "POST" "http://api.thingsee.com/v2/accounts/login" \
> -d '${
>   "email": " ",
>   "password": " "
> }'
```

```
{
  "accountAuthUuid": "82d0f2f0-8efd-11e5-b598-8bcfdb7ea12",
  "accountAuthToken": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ0cyI6MTQ0OTcyMTg5MDIxNSwidXVpZCI6IjgyZDBmMmYwLTlhZmQtMTF1NS1iNTk4LTlhY2ZkYmE3ZWEwLmIiInNjb3B1IjpbImFsbDphbGwiXSwiaWF0IjoxNDQ5NzIxODkwLCJleHAiOjE0OTAzMjY2OTB9.rUuLu_vJG"
}
```

Figure 10. Authenticating Thingsee account with REST call

Through REST calls the user is able to manage the Thingsee account and the device configurations. REST calls also provide authentication IDs for the account and device. These authentication IDs are needed if user decides to use JavaScript based nodeJS with MQTT protocol to build the server for receiving sensor measurements from Thingsee.

5.3 Google Charts API

Charts were created and used in this project to present statistics about air humidity data measured by Thingsee. The reason for choosing Google Charts was its customizable charts that come in a large variety and the simplicity of implementation.

5.3.1 Overview

Google Charts offer over 25 visualizations of data including charts, diagrams as well as histograms, maps, intervals and timelines (seen in figure 11). The service is completely free to use and the documentation can be accessed with a regular Google account. To load Google chart libraries the user needs to embed the JavaScript link in the head section of the HTML page.

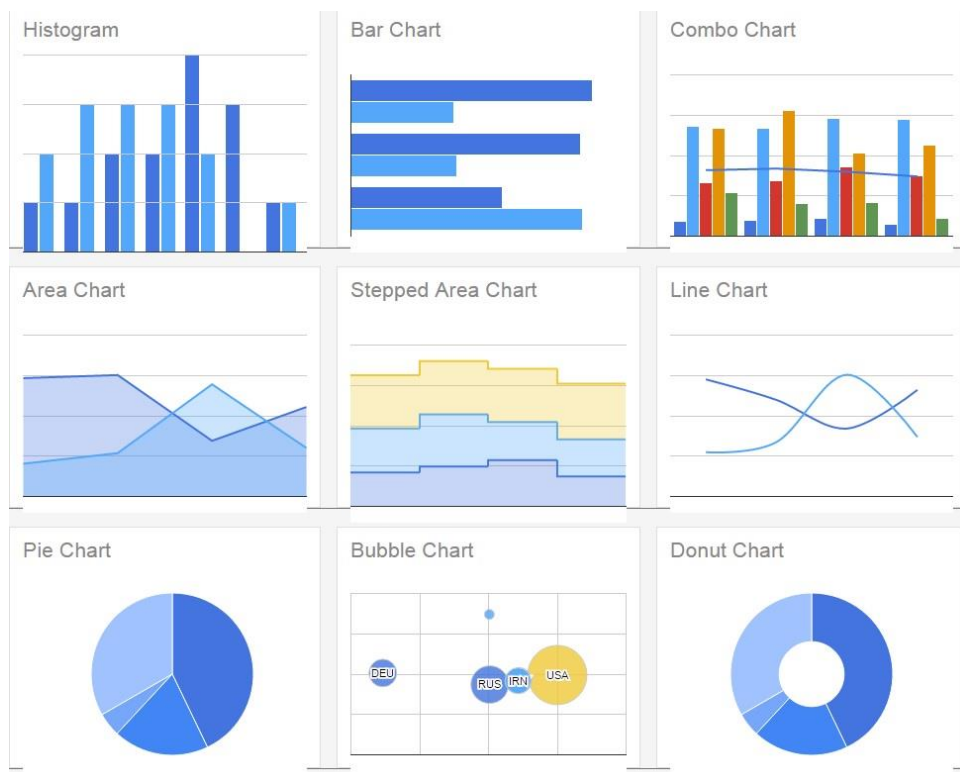


Figure 11. Variety of Charts Provided by Google Charts API. Copied from Google (2016).

The charts are customizable, interactive and portable for iPhones, iPads and Android, and require no plugins or software to work. The compatibility for cross-browsers and

platforms is achieved by rendering the charts with HTML5/SVG technology. SVG stands for Scalable Vector graphics and it defines graphics in XML format for the web. SVG graphics are scalable and don't lose quality when zoomed (W3Schools 2016.), which compliments the interactivity of google charts.

Google Charts are created as JavaScript classes that are populated with data from a data table class. The data table class can be populated from a web page, database (the method used in this research) or from a data provider that supports Chart Tools' Data-source protocol. The protocol can also be implemented on the user's own web site in case providing data to other users is an objective. As charts get their data from this class, this provides an easy way of testing which type of chart is most suitable for the user.

As good the Google Charts is for creating visual presentations of collected data, it has some data security issues. Google collects data to their own servers (Google 2014.) from some of their charts, but this can be avoided as each chart mentions this aspect in its documentation. For this research the security of the data was not an issue as no sensitive data was collected.

5.3.2 Customization

The basic libraries are ready for implementation as they do not require any configuration. Customization is an optional feature that is provided by Google, but for enhanced user experience it is worthwhile to look into. Each class has its options that are class specific that will not support other types of charts. Example of the Option object can be seen in figure 12 where the options object is given to chart.draw() function as a parameter. "is3D: true" makes a regular round pie chart appear as three dimensional.

```
chart.draw(data, {
  width: 400,
  height: 240,
  title: 'Toppings I Like On My Pizza',
  colors: ['#e0440e', '#e6693e', '#ec8f6e', '#f3b49f', '#f6c7b6'],
  is3D: true
});
```

Figure 12. Example of attributes that can be customized in a pie chart. Copied from Google (2016).

5.4 Facebook Application

The process of creating a Facebook application is quite a simple process and in many ways similar to other web development work. Facebook provides all the libraries needed and anyone can start hosting a Facebook application. Developers have to register at “Facebook for Developers” website which is also the main platform for applications and API references. Before an application can be published, it must have a working and secure URL, description and logos. Developers can implement applications both on server- and client-side with various Facebook SDKs that can be downloaded from the website.

5.4.1 Graph API

Graph API is a tool to get data in and out of Facebook's platform. It is a low-level HTTP-based API that can be used in applications to query data, post to walls, manage ads, and upload photos and other tasks. Other Facebook APIs are mostly based on the Graph API. The Graph includes various levels of information which are divided into three parts:

- nodes
- edges
- fields

Nodes represent for example pages, users and photos. Edges are the connections between them. Fields provide information such as the user's name. Graph API requests can be done with Facebook's own Graph API Explorer, directly from the console or with any language that has an HTTP library. There are different ways to query with Graph API depending on the type of data requested.

5.4.2 PHP SDK

PHP SDK is a PHP library for developers to authenticate users and make requests to the Graph API. It works as a stand-alone and can be integrated with JavaScript SDK for better user experience. Besides requests, PHP SDK allows users to upload videos and photos. It can also be used to retrieve various access tokens needed for other Facebook API implementations. PHP SDK is suitable for various tasks including building a website

with Facebook login, creating a Canvas App and page tab. PHP SDK requires PHP 5.4 or greater and mbstring extension. Composer is optional but encouraged. Installing PHP SDK without Composer involves downloading source files from the developer website and unzipping the package to the project folder. The path needs then to be added to the top of the script (seen in figure 13). (Facebook 2015.)

```
require_once __DIR__ . '/path/to/facebook-php-sdk-v4/src/Facebook/autoload.php'
```

Figure 13. Manually including autoloader inside script. Copied from Facebook (2015).

5.5 HTTP

HTTP is a request/response protocol for information systems. It is designed in a way that hides details of implementation by presenting clients an interface that is independent from any types of resources. Servers are not aware of client-side purposes and vice versa. The protocol is effective in many contexts even if the implementations evolve over time.

HTTP also acts as a way of translating communication between non-HTTP information systems. This method is called intermediation protocol. Alternative information services can reach clients through HTTP proxies and gateways, which translate their distinct protocols into hypertext format. This information can then be manipulated like HTTP services. This flexibility results in unawareness of how the protocol works behind the interface. Only the communication syntax, the desired outcome and the behaviour of recipients can be defined.

HTTP communicators are divided into client and server. Client is a software that initializes the communication by sending an HTTP request to a server. Server is also a program that manages these requests to provide clients with HTTP responses. Server and client are only terms to describe request and response actions of individual connections. The same software can act both as a client and server depending on the system. Client softwares that initiate requests (browsers, spiders, apps and command line tools) are referred to as “user agents”. Often the terms sender and recipient are also used to describe the sides of HTTP communication.

HTTP requests follow a standard format, which begins with a request-line that specifies the method, URI and protocol version. Next come header fields that contain request modifiers, information about the client, metadata and an empty line marking the end of the header. The last part is the message body that contains the payload body. The server reacts by sending an HTTP response, which starts with a status line including the protocol version, success/error code, and a comment field with explanation. Header fields are optional in HTTP responses and include the same fields as in requests. The message body is at the end of the HTTP response. (Internet Engineering Task Force 2014.)

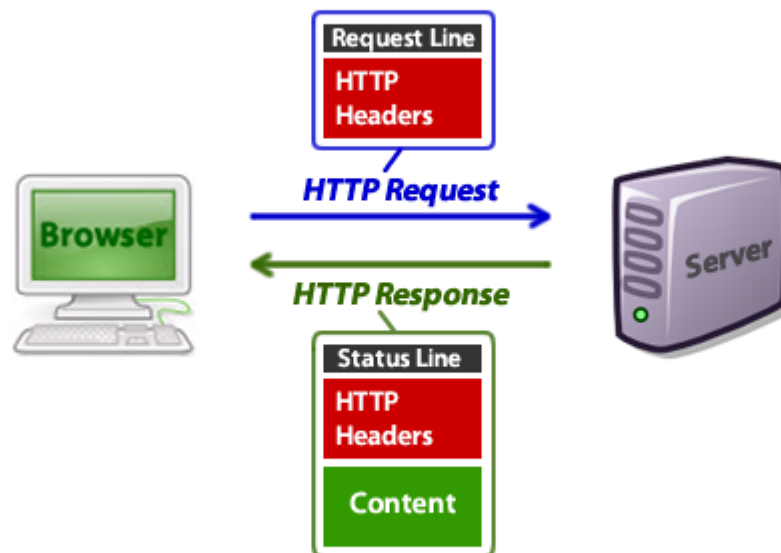


Figure 14. HTTP communication visualized. Copied from Envatotuts + (2016).

5.6 Diversity of HTTP Communication

The terms user agent and origin server are easily understood as client side browsers and websites. In reality the situation is much more diverse. HTTP user agents include household appliances, scales, stereos, apps and communication devices of many kind. Often they are running in the background and saving results to be viewed later. One

example of this behaviour are spiders that are programs designed to follow specific behaviour when crawling internet as a hypertext graph. Origin servers can also be anything from traffic cameras, office machines, and home automation units to ad selectors and video-delivery platforms. (Internet Engineering Task Force 2014)

5.7 MQTT

MQTT is a protocol specifically developed to collect data from machines. The primary purpose is to enable remote measuring, telemetry and remote monitoring. MQTT is designed to collect various data from devices that are somehow attached to the IT infrastructure. It is scalable to different sized networks and can be used with devices that need to be monitored or controlled remotely. (Schneider 2013.)

MQTT operates on hub-and-spoke architecture. Devices are connected to data concentrator server on top of TCP, which prevents data loss and makes the data stream reliable. MQTT can be used in applications to monitor thousands of sensors, which are streaming data to a single location for analysis. (Schneider 2013.)

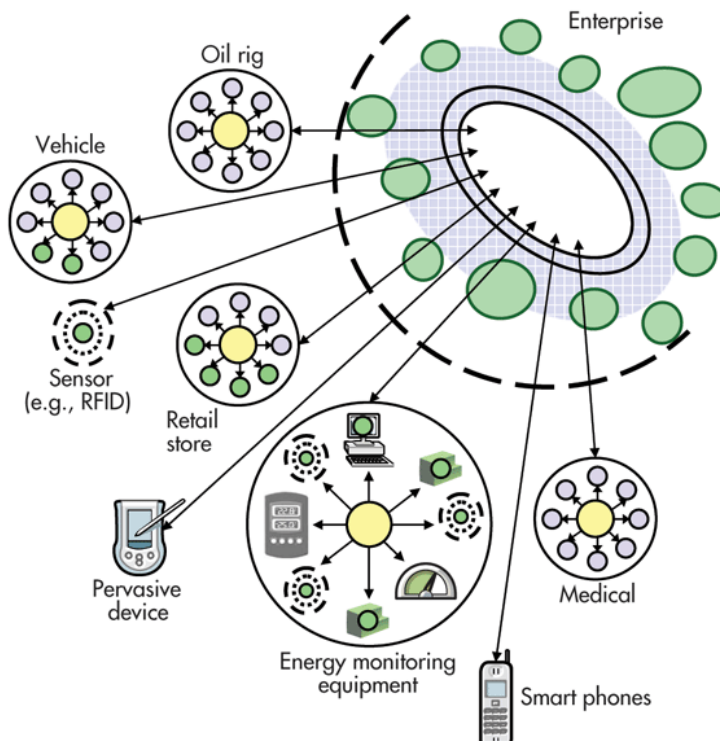


Figure 15. MQTT Hub-and-Spoke system

6 Implementation

This part of the thesis goes through the chronological order of the process it took to setup up Thingsee, Kankun, back-end and the web application to monitor the effects of the automated system on air humidity.

6.1 Thingsee Configuration

Configuration of Thingsee could be separated roughly into four parts:

- Account creation
- Wi-Fi setup
- Creation of purpose
- Redirection of data flow

Creating the Thingsee account was a straightforward process. The sign up form was filled on app.thingsee.com after which the Thingsee unit was paired with the account with step by step instructions. Thingsee Creator included instructions on how to link the device with local Wi-Fi. Web application requests the Wi-Fi information and creates a configuration file for the user to download, which is then transferred to Thingsee via a USB cable. The Wi-Fi connection to Thingsee was then verified from the device's touch screen monitor.

Once the wireless connection was setup, the Purpose was created using the instructions in Thingsee Creator. One state was created under the purpose "monitor" to measure air humidity once every hour. Redirecting the data flow was setup on the state's action tab as can be seen in Figure 16.

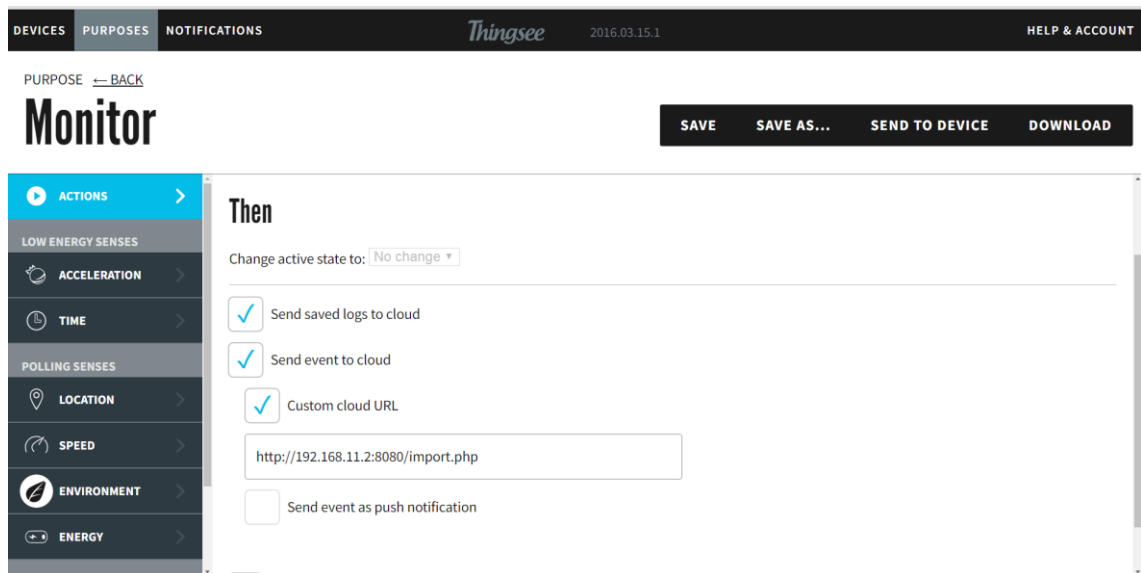


Figure 16. Redirecting Thingsee data to custom URL.

6.2 Back-end

The back-end was created using XAMPP to setup a virtual PHP host for receiving humidity measurements from Thingsee One and to provide a database for Google Charts API.

XAMPP was installed on Windows 10 with Apache and SQL servers. XAMPP installs to C drive by default and inside the xampp folder is a folder called htdocs, which acts as the root folder for the Apache server and where all project files were placed. Using the default port in Apache resulted in an error, so by editing xampp files from the GUI, the port was set to 8080, which is also indicated in the input field for the custom URL in Figure 16.

The main script in the project was called import.php, which included the following features:

- Receive humidity measurement from Thingsee as JSON
- Create a connection to SQL and save the value from parsed JSON in the database table
- Use SSH connection first to first turn humidifier off and then on depending on air humidity reading

- Algorithm counts the times script has run since the last refill and uses Facebook's PHP SDK to post a message to the air humidifier page on Facebook to inform the user when the water tank is depleted

6.2.1 Database

XAMPP's SQL support comes with phpMyAdmin, which was used to create a database and a simple table for storing the humidity measurements. The structure of the table (as seen in Figure 17) includes columns for description in case other Thingsee's sensors prove to be useful in the future applications.

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	ID	int(11)			No	None	AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
2	sensor	int(12)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
3	Kuvavus	text		Yes	NULL			Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
4	value	double			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
5	timestamp	datetime			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns

Space usage		Row statistics	
Data	57.6 KIB	Format	dynamic
Index	35 KIB	Collation	latin1_svedish_ci
Overhead	58.4 KIB	Rows	152
Effective	42.2 KIB	Row length	48 B
Total	92.6 KIB	Row size	624 B
	Optimize table	Next autoindex	2,665
		Creation	Dec 21, 2015 at 11:12 AM
		Last update	Mar 14, 2016 at 03:26 PM
		Last check	Mar 14, 2016 at 03:27 PM

Figure 17. SQL table seen in phpMyAdmin

This database was used to store measurements from Thingsee and to provide data for Google Charts API, which was used to build a monitoring interface for the user. The size of the table started to grow quickly, but the monitor only pulled the latest 25 values at any given time and no notable delays in fetching time were reported.

6.2.2 Communication with Kankun

PHP 5.6 does not come with SSH functionality, so an additional plugin was used to create a connection with the Kankun Wi-Fi switch plug. At first an attempt was made to use an ssh2 package from the PECL repository, but PHP 5.6 did not seem to support it so alternative ways were researched and phpseclib was implemented.

Phpseclib was downloaded and placed in the project folder htdocs. In import.php the plugin was attached to include_path which specifies the list of directories phpseclib also uses to look for functions. phpseclib functions read() and write() were used to execute commands on Kankun once connection was formed. (Phpseclib 2016)

```

29 set_include_path(get_include_path() . PATH_SEPARATOR . 'phpseclib');
30 include('Net/SSH2.php');
31 $ssh = new Net_SSH2('192.168.11.5');
32 if (!$ssh->login('root', 'p9z34c')) {
33     exit('Login Failed');
34 }
35 $ssh->write("echo 0 > /sys/class/leds/tp-link:blue:relay/brightness\n"); // turns the relay off
36 $output = $ssh->read('username@username:~$');
37 //if (waterlevel enough for a run){
38     if($shumValue<40){
39         $ssh->write("echo 1 > /sys/class/leds/tp-link:blue:relay/brightness\n"); // turns the relay on
40         $output = $ssh->read('username@username:~$');
41         // $current .= "x";
42         //file_put_contents($file, $current);
43     }
44     else {
45         return;
46     }

```

Figure 18. phpseclib script used to connect to Kankun

6.2.3 Water-level Counter

Keeping track of the water level consisted of a couple of code snippets in import.php, which counted the amount of times phpseclib was used to turn the humidifier on by keeping count in a text file. After the humidifier had been turned on, the application appended an “x” to waterlevel.txt, which was checked every time the application started. If the program detected that the file contained enough symbols, it did not turn on the humidifier and proceeded to send a Facebook notification to the user.

6.2.4 Facebook Notification

A Facebook application was created on the Facebook Developers page and paired with a Facebook page “Air humidifier” which served as the notification platform for the message.

In general, creating a Facebook application includes several steps. First, the developer has to pick a name, category and sub-category, which should be specified correctly if the application is released for public use. For certain integrations Facebook requires the developer to submit the application for a review before it can be published. (Facebook 2016.)

On Facebook for Developers website on the settings tab a page was added as a platform and Facebook page “humidifier” was linked with the application. Graph API Explorer was used to acquire an access token to be used in the PHP application.

When the length of waterlevel.txt reached the limit, Facebook API’s PHP SDK was used to send a status post to the Facebook page. The connection requires App ID, API Version and App secret, which were copied from the Dashboard view in Application settings. The post was sent to the page as an HTTP post, which was confirmed from Facebook. (see Figures 19 and 20)

```
$response = $fb->post('/me/feed?access_token=*****&message=More Water!');
```

Figure 19. Sending a Facebook post with Facebook API.

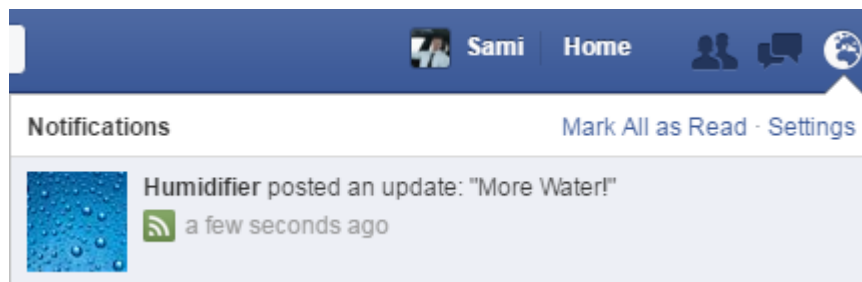


Figure 20. Confirming successful post.

6.3 Web App Monitor

A web app monitor was built using Google Charts API by providing data from SQL in JSON format. Export.php was written to fetch echo data in JSON based on which “?type=” chart.js was requesting. This was done in case other Thingsee sensors would be used in the future. Chart.js consisted of three functions:

1. Building the chart
2. Getting latest measurement
3. init() function which ran two first ones

Functions 1 and 2 were given a div to append to and a dataUrl, which pointed to export.php?type=hum to tell export.php to fetch 25 latest humidity related entries from SQL. Limit was set to 25 so that the chart would show only the last 25 hours in the graph.

The web application also needed a simple, user friendly way to monitor and reset the water level counter once the water was depleted and the user had refilled the tank. Percentage counter and reset button were added in index.php. The complete UI can be seen in figure 21.

Humidity Monitor

Latest Value

Humidity: 32.2 %

Water tank level at 0%

Reset:

Measurements from the last 25 hours

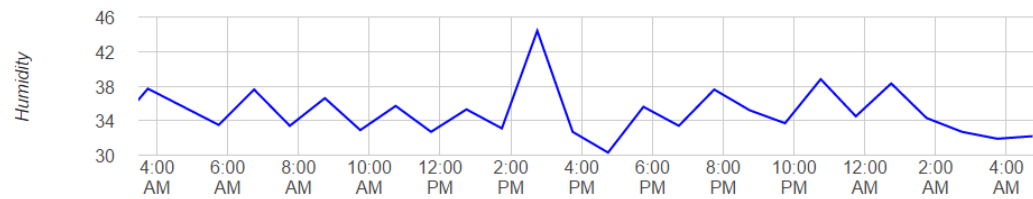


Figure 21. Web app monitor

7 Testing

During testing, Thingsee was placed approximately 1 metre behind the humidifier in a way that the steam was emitted in opposite direction. This was done to minimize the chance of incorrect measurements. During testing, the room was used for normal living to simulate a realistic use-case scenario.

Testing was divided into three phases. In the first phase average air humidity and spread were measured for 24 hours without the humidifier. In the second phase the air humidifier was kept turned on until the water tank ran empty. In the third phase the rest of the system was used and kept on until tank had to be refilled. Air humidity and water tank life time were observed and the results were compared to phase 2. The emission level was kept the same for phases 2 and 3.

The hypothesis was that during phase 1 air humidity would vary depending on external factors such as weather conditions and time of day. In phase 2 these factors should increase the lowest values and highlight the highest values compared to phase 1. Phase 3 should last longer than phase 2 and the air humidity should stay in the recommended range of 30-40% at all times. The graph of phase 3 should look like a saw tooth, in which humidity rises when the humidifier is turned on and goes down when it is off.

8 Discussion and Evaluation

8.1 Results

Humidity was little under 30% in the start of phase 1. During the first 8 hours no significant change was recorded. Around 6 am humidity started to slowly drop for 4 hours and dipped to 23%. For the next 5 hours it kept slowly rising to 27%. Thingsee failed to send data between 2:45pm - 5:45pm, because battery had run out despite being plugged to a charger. This might have caused the next measurement to dip under 20%. Haltian states that Thingsee might require few measurements after restart to calibrate the sensors. Humidity raised back to 25% in a few hours and kept rising to 30% where it stagnated until the end of the test. Average humidity during phase 1 was 26.9%.

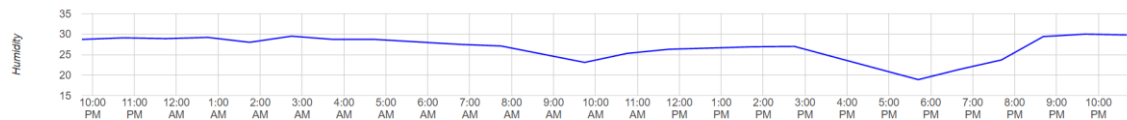


Figure 22. Phase 1

In the start of phase 2 humidity was measured at 30%. It kept rising for three hours until stagnating at ~35%. The stagnation period lasted for 9 hours, which was followed by another 3-hour rising period. After 15 hours from the start humidity peaked at 42%. After the peak, humidity stagnated again and stayed around 40-41% for six hours until the water tank depleted. After depletion humidity decreased to 30% in 9 hours. The humidifier kept humidity in range of 30-40% with one water tank for 22 hours. Average measured humidity from that time was 35.19%.

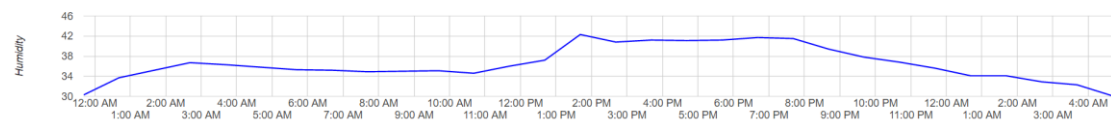


Figure 23. Phase 2

Phase 3 was started when humidity had decreased to 30% to match the conditions of the second phase. Humidity started to rise for 8 hours until it reached over 35%. During the following 15 hours, humidity kept rising and falling in a predictable pattern, while staying in the desired range. After the water tank depleted humidity decreased to 30% in

4 hours. IoT powered humidifier system kept air humidity in the desired range for 39 hours. Average humidity during this phase was ~34%.

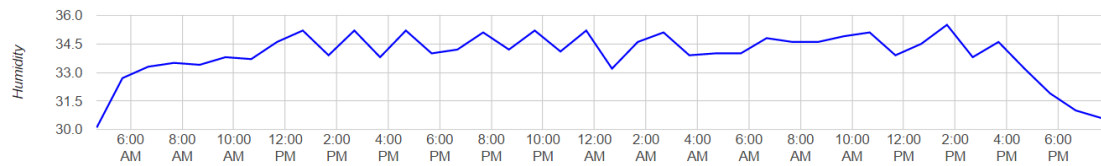


Figure 24. Phase 3

8.2 Comparison

The results showed that a humidifier powered with IoT technology can help keep humidity at desired levels over 77% longer than without it. The average humidity of phase 3 was ~1.2% lower than in phase 2.

Time seemed to have an effect on humidity. In phase 1 humidity started to drop around sunrise (6:47) and went back up around sundown (20:04). This evidence is supported by the stagnation of humidity in phase 2 during the same time span. Evidence from phase 3 suggests that this effect is strongest during 6 am-12 pm.

8.3 Evaluation

Although the system built in this project proved to provide optimization for the air humidifier, there are topics that require attention if research is continued.

8.3.1 The task of the water humidifier

The project optimized and automated water humidifiers task, but the water tank had to be still refilled manually by removing it from the humidifier and filling it in the bathroom. Future applications should consider attaching a small scale water pump between the humidifier and another Kankun unit, which would be synced together in the PHP script. Water pump would provide constant flow of water to tank making the system fully automated. This type of system should be designed in a way that allows the water pump to draw water from the tap or washing machine coupling.

Adjustment of emission level is important for test results. In this research the emission level knob was turned approximately to 60%, which caused humidity to rise over 40% for hours. A higher level would hypothetically have a bigger effect causing air humidity to stay at unwanted levels for even longer. Comparisons should be made about the benefits of different emission levels.

8.3.2 Test Area

The water humidifier used in this study is usable up to 125 square meters. During this study it was placed in a 40 square meter apartment, but measurements were only taken inside the 10 square meter room where it was located. Future configurations should take other rooms into consideration as well, hypothesis being that it takes longer time for humidity to evenly settle around an apartment. Several humidifiers and sensors should be used to achieve best results.

The project was conducted in a student housing where electricity is free for students. A 24/7 online system that requires the PC to be on at all times would cause spikes in electricity bills in normal apartments. In these scenarios it might be economically wise to consider switching to Arduino or another similar electronic platform to run the server. These solutions also offer their own air humidity sensors, which would further increase the benefits from implementing them into these types of projects.

Tests showed that external factors such as weather conditions and time of day have an effect on air humidity. All further tests should run the phases simultaneously to assure identical conditions for the measurements.

8.3.3 Thingsee

Thingsee performed as intended for most of the time. During early tests occasional data loss was reported. In cases of data loss Thingsee had measured the values but the data never reached the server. All these occurrences were reported during the time the PC running the server was in sleep mode, and the problem was fixed when the sleep mode was turned off. However, some data got through even on sleep mode.

Haltian promises Thingsee up to one year of battery life between recharging, but specific use-case scenarios to achieve these results were not found (Thingsee 2016). In this study Thingsee was sending one measurement every hour, which depleted the battery in approximately four days. For longer testing periods this is not a problem, because Thingsee can be charged while it sends data. However, Thingsee did run out of battery once during phase 1 despite being attached to charger. Suspected reason for this was a bad contact with the charger.

8.3.4 Security

The IT security of Kankun can be debatable, but no attacks or hacking was reported during the testing phase. If such an attack would happen, the hacker would gain control of the Kankun and turn the relay on and off whenever and how often as possible. In this research the worst case scenario would have been uncontrolled levels of air humidity. This type of security risk can be addressed by changing the username and password used in the SSH connection to the Kankun.

9 Conclusion

The goal of this thesis was to give an overview of IoT as a technology and showcase its capabilities. The project successfully managed to optimize an air humidifier's activity by providing the user with more control over humidity levels and increasing its run time between refills. The results showed that the IoT technology can be used to bring improvements to a household appliance with minimal amount of hardware.

In further tests the accuracy of the results should be improved by running all phases simultaneously. This would require a bigger test area with multiple rooms with identical conditions. Notable external factors were weather conditions and time of day.

The system brought optimization, but did not fully make the humidifier's operation automatic. This could be achieved with an automated water refill system that would provide a constant water source for the humidifier.

References

Ashton, Kevin. That 'Internet of Things' Thing [online]. 2009.

URL: <http://www.rfidjournal.com/articles/view?4986>. Accessed 23 March 2016.

Burgener, Sullivan. Hacking Kankun Smart Wifi Plug [online]. Anites. 2015.

URL: <http://www.anites.com/2015/01/hacking-kankun-smart-wifi-plug.html>.

Accessed 23 March 2016.

Cambridge. The Trojan Room Coffee Machine [online]. 2011.

URL: <https://www.cl.cam.ac.uk/coffee/coffee.html>.

Accessed 23 March 2016.

CCHRC. Indoor Air Quality [online]. Cold Climate Housing Research Centre. 2014.

URL: <http://www.cchrc.org/indoor-air-quality>. Accessed 23 March 2016.

Freescale. What the Internet of Things (IoT) Needs to Become a Reality [online]. 2016.

URL: https://www.digikey.com/Web%20Export/Supplier%20Content/Freescale_375/PDF/freescale-internet-of-things-reality.pdf?redirected=1.

Accessed 23 March 2016.

Envatotuts+. HTTP Headers for Dummies [online]. 2016.

URL: <http://code.tutsplus.com/tutorials/http-headers-for-dummies--net-8039>.

Accessed 23 March 2016.

Evans, Dave. The Internet of Things: How the Next Evolution of the Internet Is Changing Everything [online]. Cisco. 2011.

URL: http://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf. Accessed 23 March 2016.

Facebook. App Review [online]. 2016.

URL: <https://developers.facebook.com/docs/apps/review>.

Accessed 23 March 2016.

Facebook. Getting started with the Facebook SDK for PHP [online]. 2015.

URL: <https://developers.facebook.com/docs/php/gettingstarted>.

Accessed 23 March 2016.

Gartner. Gartner's 2014 Hype Cycle for Emerging Technologies Maps the Journey to Digital Business [online]. 2014.

URL: <http://www.gartner.com/newsroom/id/2819918>.

Accessed 23 March 2016.

Gold-Group. Gartner's Hype Cycle of Emerging Technologies [online]. 2015

URL: <http://gold-group.com/2013/10/22/qr-codes-beyond-hype-cycle/>.

Accessed 23 March 2016.

Google. Chart Gallery [online]. 2016.

URL: <https://developers.google.com/chart/interactive/docs/gallery>.

Accessed 23 March 2016.

Google. Google APIs Terms of Service [online]. 2014.

URL: <https://developers.google.com/terms/>.

Accessed 23 March 2016.

Google. Google Trends [online]. 2016.

URL: <http://www.google.com/trends/explore#q=internet%20of%20things>.

Accessed 23 March 2016.

Google. How to Customize Charts [online]. 2016.

URL: https://developers.google.com/chart/interactive/docs/customizing_charts.

Accessed 23 March 2016.

Hanumanthappa.J, Manjaiah D.H. A Study on Comparison and Contrast between IPv6 and IPv4 Feature sets [online]. 2008.

URL:https://www.researchgate.net/publication/209423385_A_Study_on_Comparison_and_Contrast_between_IPv6_and_IPv4_Feature_Sets.

Accessed 23 March 2016.

International Telecommunication Union. Y.2060 [online]. 2012

URL: <http://handle.itu.int/11.1002/1000/11559-en?locatt=for-mat:pdf&auth>.

Accessed 23 March 2016.

Internet Engineering Task Force. Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing [online]. Internet Engineering Task Force. 2014.

URL: <http://tools.ietf.org/html/rfc7230>. Accessed 23 March 2016.

IoT6. IPv6 Advantages for IoT [online]. 2014.

URL: http://iot6.eu/ipv6_advantages_for_iot.

Accessed 23 March 2016.

MLove. Thingsee One [online]. MLove. 2016.

URL: <http://confestival.mlove.com/expo-thingsee-by-kii/>.

Accessed 23 March 2016.

One2more. Wireless Wi-Fi Smart Plug Kankun [online]. One2more. 2016

URL: <http://www.one2more.com/en/798-wireless-wi-fi-smart-plug-kankun-kk-sp3-wireless-remote-control-app-for-android-ios-us-ac90-265v.html>

Accessed 23 March 2016.

Phpseclib. phpseclib: SSH2 Examples and Notes [online]. SourceForge. 2016.

URL: <http://phpseclib.sourceforge.net/ssh/examples.html>.

Accessed 23 March 2016.

PTC. IoT Use Cases: Start Your Connected Journey Here [online].

URL: <http://www.ptc.com/File%20Library/IoT/IoT-Use-Case-eBook.pdf>

Accessed 23 March 2016.

Schneider, Stan. Understanding The Protocols Behind The Internet of Things [online].
Electronic Design. 2013.
URL: <http://electronicdesign.com/iot/understanding-protocols-behind-internet-things>.
Accessed 23 March 2016.

Thingsee. Data Sheet [online]. 2016.
URL: <https://thingsee.com/data-sheet>.
Accessed 23 March 2016.




















































Thingsee. How do I conserve battery power? [online]. 2016.
URL: <https://thingsee.zendesk.com/hc/en-us/articles/203593742-How-do-I-conserve-battery-power>-
battery-power-
Accessed 23 March 2016.























































Wilfa. Ilmankostutin Sky HU-6W [online]. Wilfa. 2015.
URL: <http://www.wilfa.fi/tuotteet/huoneilmanhoito/ilmankostutin/sky/#>.
Accessed 23 March 2016.

W3Schools. SVG Tutorial [online]. 2016.
URL: <http://www.w3schools.com/svg/>.
Accessed 23 March 2016.























































XAMPP. About [online]. 2016.
URL: <https://www.apachefriends.org/about.html>
Accessed 23 March 2016.

Appendix 1. Test Measurements

		ID	sensor	Kuvaus	value	timestamp	▲ 1
<input type="checkbox"/>	 Edit  Copy  Delete	2872	0	Humidity	30.3	2016-03-30 22:41:36	
<input type="checkbox"/>	 Edit  Copy  Delete	2873	0	Humidity	33.7	2016-03-30 23:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2874	0	Humidity	35.2	2016-03-31 00:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2875	0	Humidity	36.7	2016-03-31 01:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2876	0	Humidity	36.3	2016-03-31 02:41:06	
		ID	sensor	Kuvaus	value	timestamp	▲ 1
<input type="checkbox"/>	 Edit  Copy  Delete	2877	0	Humidity	35.8	2016-03-31 03:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2878	0	Humidity	35.3	2016-03-31 04:41:36	
<input type="checkbox"/>	 Edit  Copy  Delete	2879	0	Humidity	35.2	2016-03-31 05:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2880	0	Humidity	34.9	2016-03-31 06:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2881	0	Humidity	35	2016-03-31 07:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2882	0	Humidity	35.1	2016-03-31 08:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2883	0	Humidity	34.6	2016-03-31 09:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2884	0	Humidity	36	2016-03-31 10:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2885	0	Humidity	37.2	2016-03-31 11:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2886	0	Humidity	42.3	2016-03-31 12:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2887	0	Humidity	40.8	2016-03-31 13:41:06	
<input type="checkbox"/>	 Edit  Copy  Delete	2888	0	Humidity	41.2	2016-03-31 14:41:07	

<input type="checkbox"/>	 Edit	 Copy	 Delete	2889	0	Humidity	41.1	2016-03-31 15:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2890	0	Humidity	41.2	2016-03-31 16:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2891	0	Humidity	41.7	2016-03-31 17:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2892	0	Humidity	41.5	2016-03-31 18:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2893	0	Humidity	39.4	2016-03-31 19:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2894	0	Humidity	37.8	2016-03-31 20:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2895	0	Humidity	36.8	2016-03-31 21:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2896	0	Humidity	35.6	2016-03-31 22:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2897	0	Humidity	34.1	2016-03-31 23:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2898	0	Humidity	34.1	2016-04-01 00:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2899	0	Humidity	32.9	2016-04-01 01:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2900	0	Humidity	32.3	2016-04-01 02:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2901	0	Humidity	30.1	2016-04-01 03:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2902	0	Humidity	32.7	2016-04-01 04:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2903	0	Humidity	33.3	2016-04-01 05:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2904	0	Humidity	33.5	2016-04-01 06:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2905	0	Humidity	33.4	2016-04-01 07:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2906	0	Humidity	33.8	2016-04-01 08:41:07

Console

<input type="checkbox"/>	 Edit	 Copy	 Delete	2907	0	Humidity	33.7	2016-04-01 09:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2908	0	Humidity	34.6	2016-04-01 10:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2909	0	Humidity	35.2	2016-04-01 11:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2910	0	Humidity	33.9	2016-04-01 12:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2911	0	Humidity	35.2	2016-04-01 13:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2912	0	Humidity	33.8	2016-04-01 14:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2913	0	Humidity	35.2	2016-04-01 15:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2914	0	Humidity	34	2016-04-01 16:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2915	0	Humidity	34.2	2016-04-01 17:41:07
<input type="checkbox"/>	 Edit	 Copy	 Delete	2916	0	Humidity	35.1	2016-04-01 18:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2917	0	Humidity	34.2	2016-04-01 19:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2918	0	Humidity	35.2	2016-04-01 20:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2919	0	Humidity	34.1	2016-04-01 21:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2920	0	Humidity	35.2	2016-04-01 22:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2921	0	Humidity	33.2	2016-04-01 23:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2922	0	Humidity	34.6	2016-04-02 00:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2923	0	Humidity	35.1	2016-04-02 01:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2924	0	Humidity	33.9	2016-04-02 02:41:08
<input type="checkbox"/>	Console							

<input type="checkbox"/>	 Edit	 Copy	 Delete	2925	0	Humidity	34	2016-04-02 03:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2926	0	Humidity	34	2016-04-02 04:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2927	0	Humidity	34.8	2016-04-02 05:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2928	0	Humidity	34.6	2016-04-02 06:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2929	0	Humidity	34.6	2016-04-02 07:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2930	0	Humidity	34.9	2016-04-02 08:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2931	0	Humidity	35.1	2016-04-02 09:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2932	0	Humidity	33.9	2016-04-02 10:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2933	0	Humidity	34.5	2016-04-02 11:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2934	0	Humidity	35.5	2016-04-02 12:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2935	0	Humidity	33.8	2016-04-02 13:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2936	0	Humidity	34.6	2016-04-02 14:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2937	0	Humidity	33.2	2016-04-02 15:41:38
<input type="checkbox"/>	 Edit	 Copy	 Delete	2938	0	Humidity	31.9	2016-04-02 16:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2939	0	Humidity	31	2016-04-02 17:41:08
<input type="checkbox"/>	 Edit	 Copy	 Delete	2940	0	Humidity	30.6	2016-04-02 18:41:08