RIOT in Internet of Things

The Mini-Loon Project

Agenda

1. Team Introduction
2. Explaining the background of the project
3. Our Goals and Requirements
4. Prototype Showcase
5. Next Steps

Team Projekt Management

Sofia Knap
Eneida Koltraka

Team Balloon - Control

Tobias Westphal
Johannes Nodop
Karl Klemann
Michael Mylius
Lasse Rosenow

Team Gateway - Cloud

Katerina Milenkovski
Tristan Ropers
Lasse Prüß

Team Smartphone-App

Diogo Henriques
Diogo Chumbo
Bruno Rodrigues

Background

Loon LLC is an Alphabet Inc. subsidiary working on providing Internet access to rural and remote areas. The company uses high-altitude balloons in the stratosphere at an altitude of 18 km to 25 km to create an aerial wireless network with up to 1 Mbit/s speeds.

Disaster Scenario

of Ginger Zee/ABC News

Our Goal

Use IoT suited balloons to detect drastic changes in weather conditions, to help prevent damages in case of natural disasters.

Example Use Case

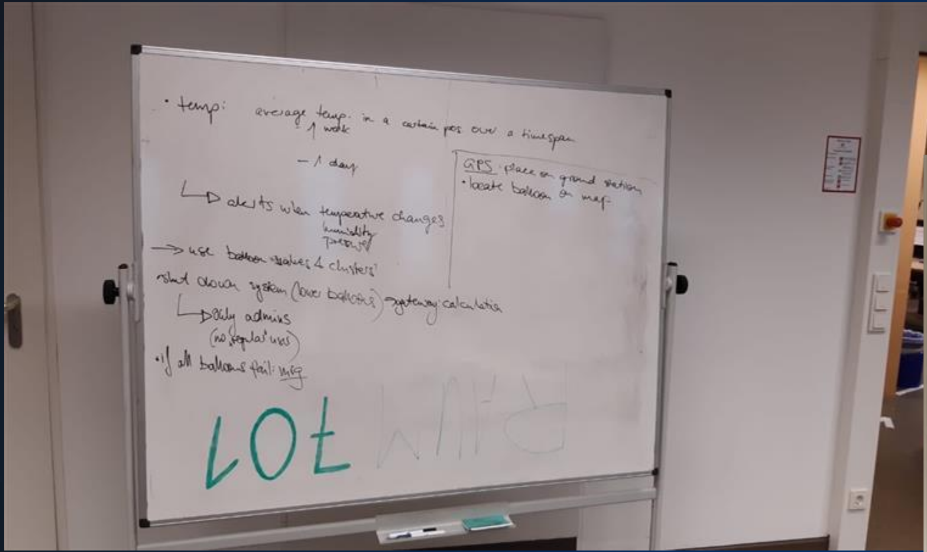
Temperature

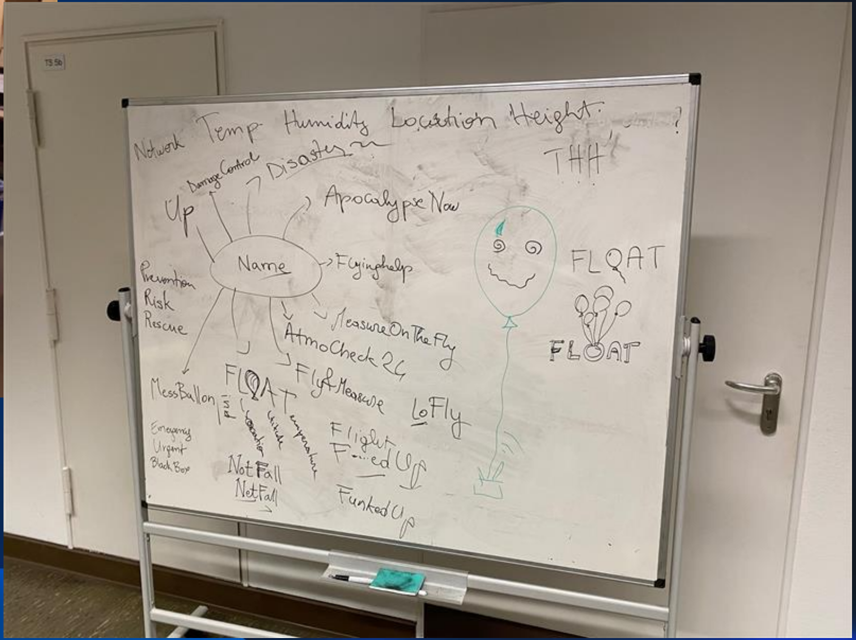
High temperatures detected

Humidity

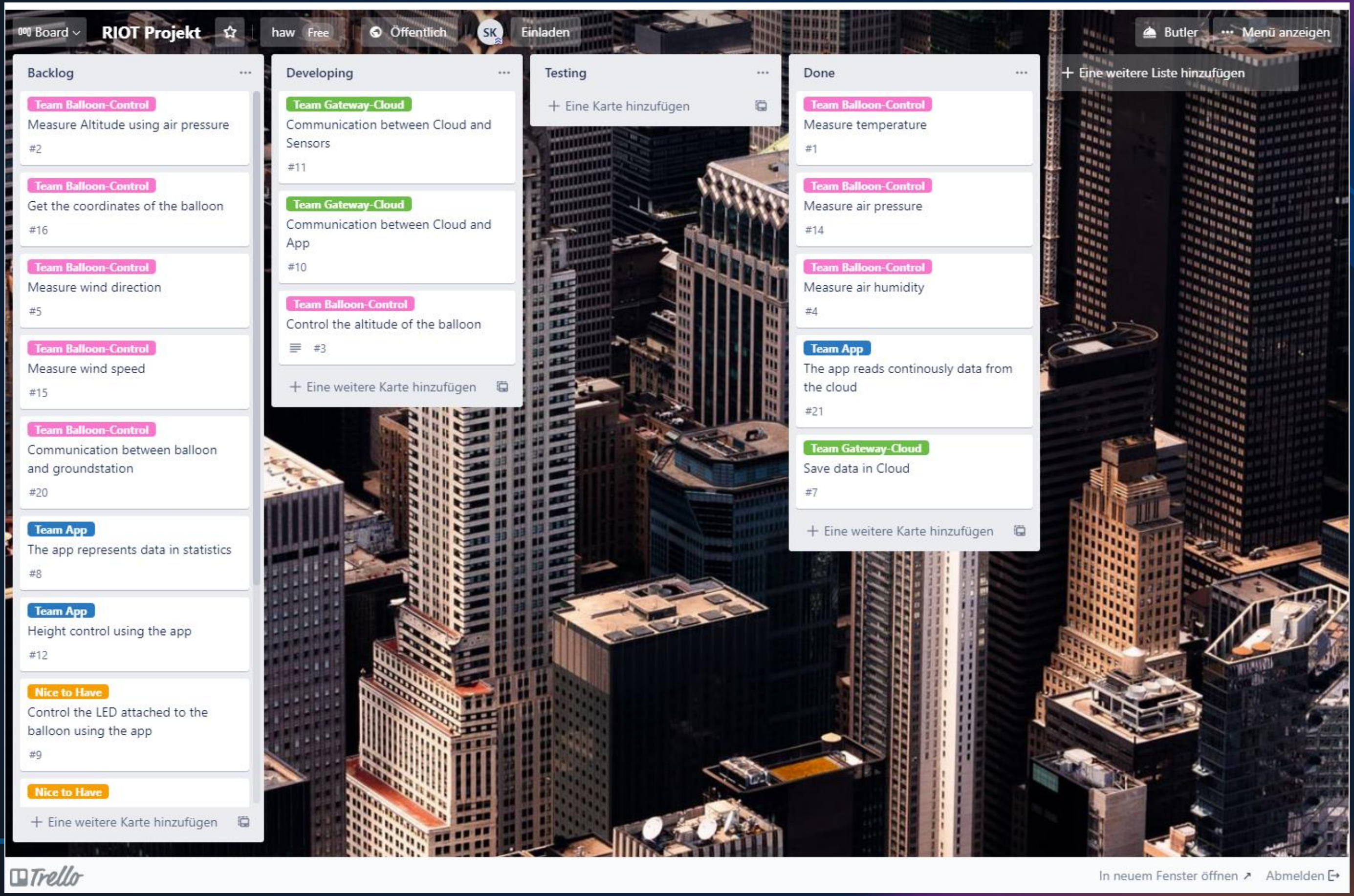
Low humidity detected

Brainstorming


• temp: average temp. in a certain pos over a timespan
1 week
– 1 day
↳ alerts when temperature changes
humidity
pressure
→ use balloon values 4 clusters?
shut down system (over balloons) gateway calculation
↳ Daily admins
(no "regular" user)
• If all balloons fail: msg
GPS: place on ground station
• locate balloon on map
LOT



Requirements Engineering



Backlog

- Team Balloon-Control
  Measure Altitude using air pressure
  #2
- Team Balloon-Control
  Get the coordinates of the balloon
  #16
- Team Balloon-Control
  Measure wind direction
  #5
- Team Balloon-Control
  Measure wind speed
  #15
- Team Balloon-Control
  Communication between balloon and groundstation
  #20
- Team App
  The app represents data in statistics
  #8
- Team App
  Height control using the app
  #12
- Nice to Have
  Control the LED attached to the balloon using the app
  #9
- Nice to Have

Developing

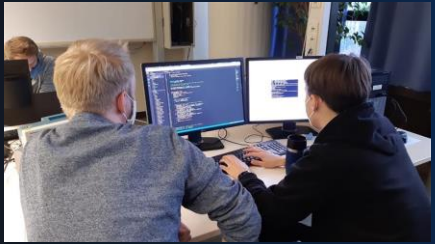
- Team Gateway-Cloud
  Communication between Cloud and Sensors
  #11
- Team Gateway-Cloud
  Communication between Cloud and App
  #10
- Team Balloon-Control
  Control the altitude of the balloon
  #3

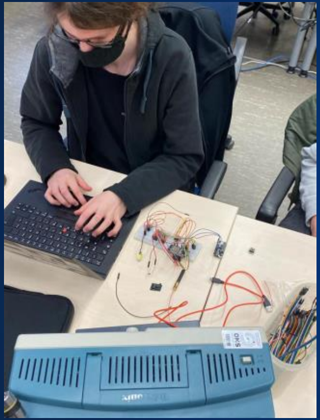
Testing

Done

- Team Balloon-Control
  Measure temperature
  #1
- Team Balloon-Control
  Measure air pressure
  #14
- Team Balloon-Control
  Measure air humidity
  #4
- Team App
  The app reads continously data from the cloud
  #21
- Team Gateway-Cloud
  Save data in Cloud
  #7

Work in Progress …





Desperation…

Balloon-Control

In depth

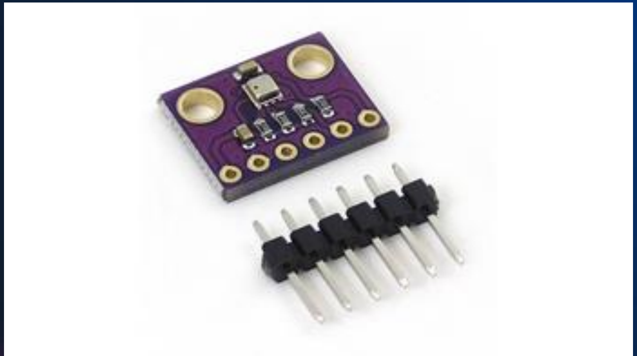
struct balloon_control {

- BME280
- GNSS module
- Valve
- Mini-Loon-Board
    - Electric Circuit
    - Footprints
- Reading sensor values with SAUL and communicating via CoAP
- LoRa-WAN & CBOR

}

BME280 sensor

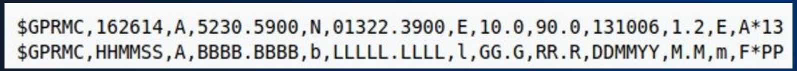
- Sensor for measuring relative humidity, pressure, and temperature
- Low power consumption (3.6 µA @ 1 Hz (H, P, T) / 0.1 µA in sleep mode)
- Compact and lightweight
- Broad operation range (300…1100 hPA; - 40…85°C)
- Sensor values can be read with SAUL



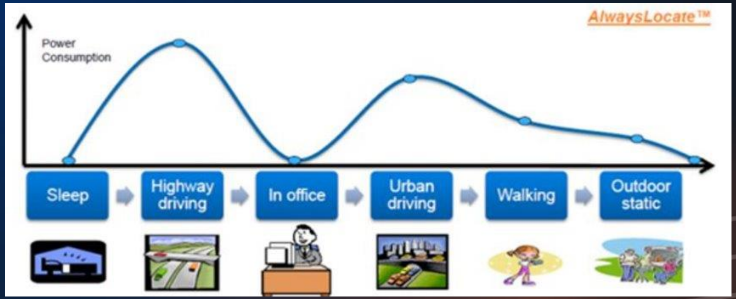
GNSS module (Quectel L96)

- Receiver types: NAVSTAR GPS, GLONASS, Galileo and BeiDou
- Receive data: NMEA standard
- Exchange data with ESP32: UART Interface
- Modes: Full on (20-25mA), GLP(Ø 10-20mA), AlwaysLocate(Ø 2.7mA)
- Offers longitude/latitude position, velocity, date and time

NMEA Output:

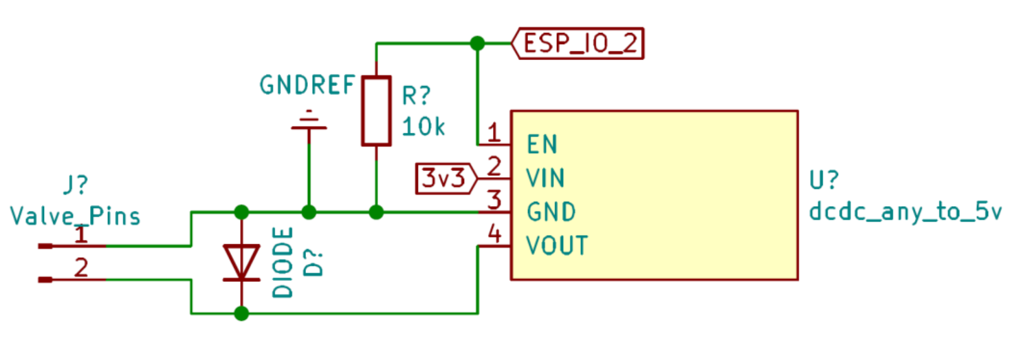


```
$GPRMC,162614,A,5230.5900,N,01322.3900,E,10.0,90.0,131006,1.2,E,A*13
$GPRMC,HHMMSS,A,BBBB.BBBB,b,LLLLL.LLLL,l,GG.G,RR.R,DDMMYY,M.M,m,F*PP
```



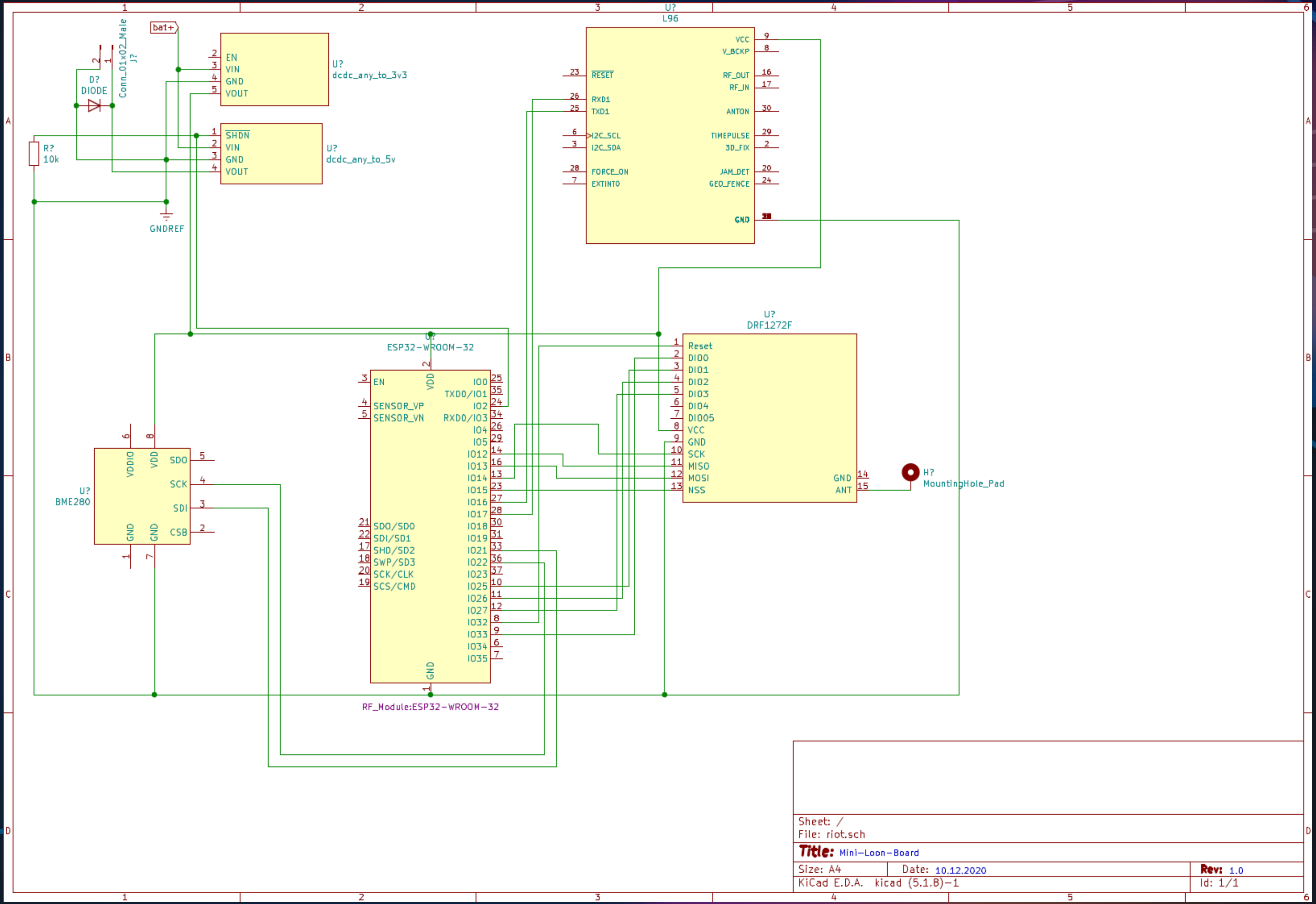
Valve

- Release helium to let the balloon sink
- Needs at least 5V
- Will be enabled via the “enable pin” on the DC/DC converter
- By default, the “enable pin” is pulled up
  - 10k ohm pull-down resistor (causes a small, unnecessary energy drain)
  - Alternative: remove pull-up from the DC/DC converter board

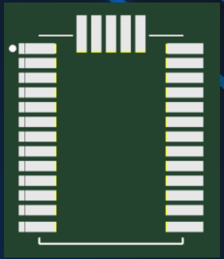


Mini-Loon-Board:

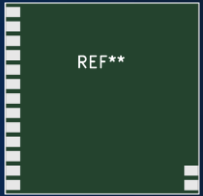
Electric Circuit

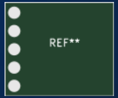


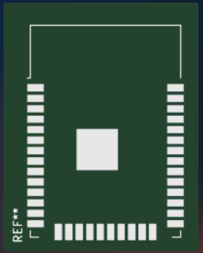
Mini-Loon-Board: Footprints



- The sockets for our sensors, chips etc.
- Will be put on a custom board, which will be produced by a third-party company
- The sensors etc. only need to be soldered on to their own socket
- No need of cables
- Only one board for all our chips
- Way less overall weight then the prototype







Reading sensor values with SAUL and communicating via CoAP

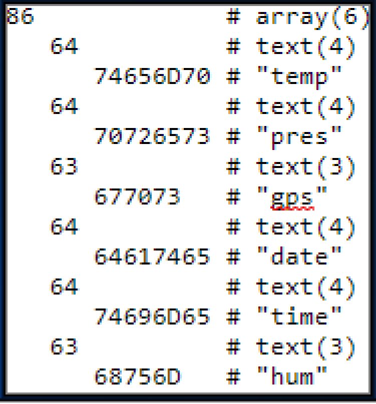
- [Co]nstrained [A]pplication [P]rotocol
- [S]ensor [A]ctuator [U]ber [L]ayer
- Create CoAP resources for temperature, humidity and air pressure
    - /sens/temp -> temperature
    - /sens/hum -> humidity
    - /sens/press -> air pressure

- „coap get“-Request triggers SAUL sensor reading for resource
    - Example: „coap get <Server-IP> <Port> /sens/temp“ triggers temperature measurement

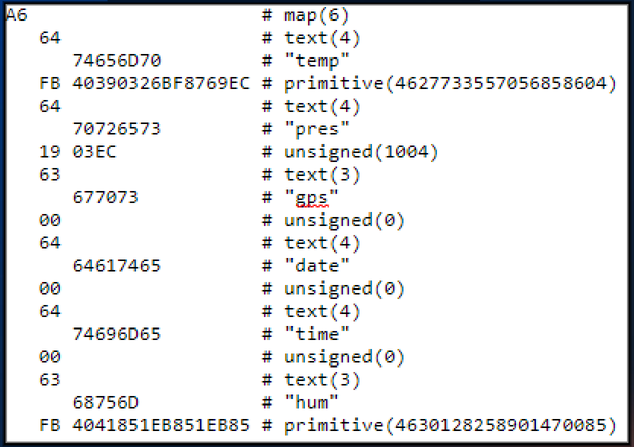
LoRa-WAN & CBOR

- [Lo]ng-[Ra]nge-[W]ide [A]rea [N]etwork
  - Low Power usage
  - Communication over gateways
  - Schedules requests
  - Downlink only after Uplink
- [C]oncise [B]inary [O]bject [R]epresentation
  - Less data exchange
  - Encoding data as hex
  - Decoding hex to data

["temp", "pres", "gps", "date", "time", "hum"]



```
86                  # array(6)
   64               # text(4)
      74656D70      # "temp"
   64               # text(4)
      70726573      # "pres"
   63               # text(3)
      677073        # "gps"
   64               # text(4)
      64617465      # "date"
   64               # text(4)
      74696D65      # "time"
   63               # text(3)
      68756D        # "hum"
```

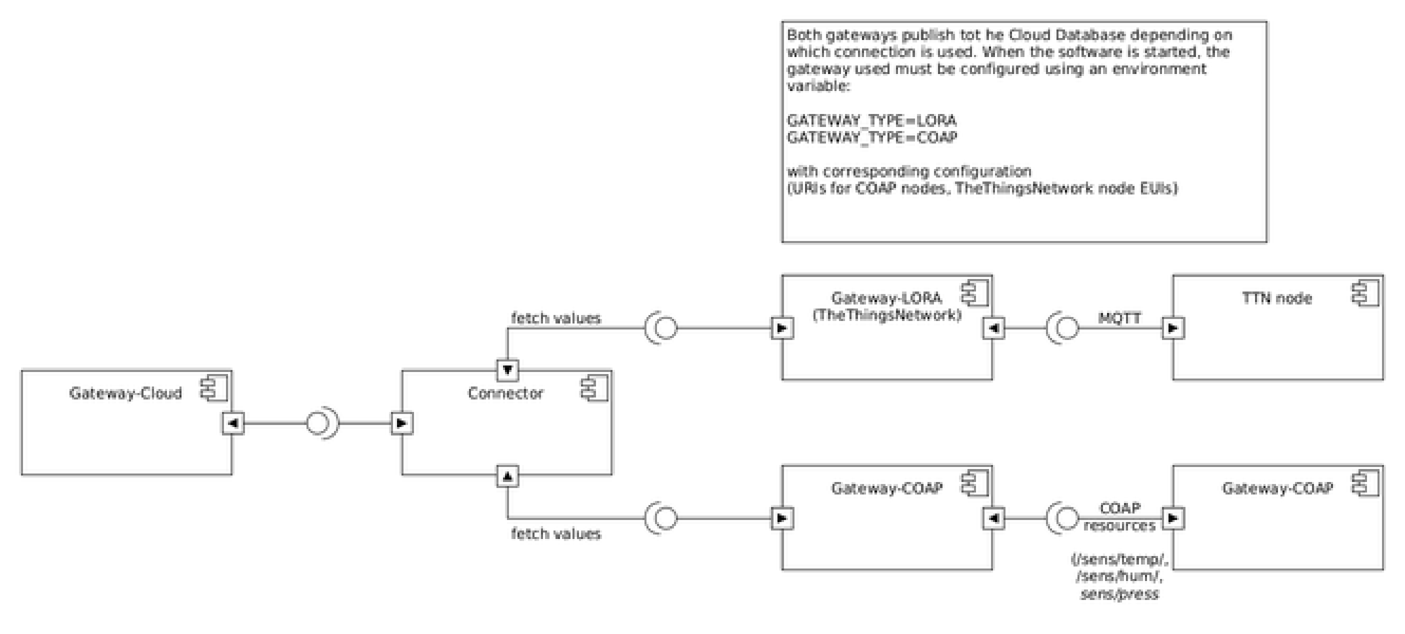


```
A6                        # map(6)
   64                     # text(4)
      74656D70            # "temp"
   FB 40390326BF8769EC    # primitive(4627733557056858604)
   64                     # text(4)
      70726573            # "pres"
   19 03EC                # unsigned(1004)
   63                     # text(3)
      677073              # "gps"
   00                     # unsigned(0)
   64                     # text(4)
      64617465            # "date"
   00                     # unsigned(0)
   64                     # text(4)
      74696D65            # "time"
   00                     # unsigned(0)
   63                     # text(3)
      68756D              # "hum"
   FB 4041851EB851EB85    # primitive(4630128258901470085)
```

{"temp": 25.01231, "pres": 1004, "gps": 0, "date": 0, "time": 0, "hum": 35.04}

Gateway-Cloud

- Gateway handles the communication between the app and the balloons
- TheThingsNetwork (LORA) and COAP are supported communication protocols
- Coordination of multiple balloons in clusters



Both gateways publish tot he Cloud Database depending on which connection is used. When the software is started, the gateway used must be configured using an environment variable:

GATEWAY_TYPE=LORA
GATEWAY_TYPE=COAP

with corresponding configuration
(URIs for COAP nodes, TheThingsNetwork node EUIs)

Gateway-Cloud Current State

- COAP prototype for transferring Temperature values is working
- Temperature, humidity and pressure values fetched over COAP
- Values stored in Firebase for App to use

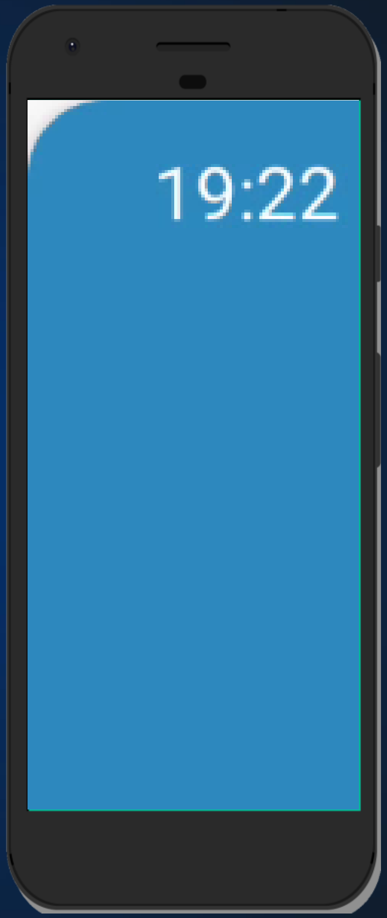
Gateway-Cloud Technologies Used

- Golang as programming language
- go-coap (https://github.com/plgd-dev/go-coap)
- Google Firebase (https://firebase.google.com/)
- The Things Network Go SDK (https://github.com/TheThingsNetwork/go-app-sdk)

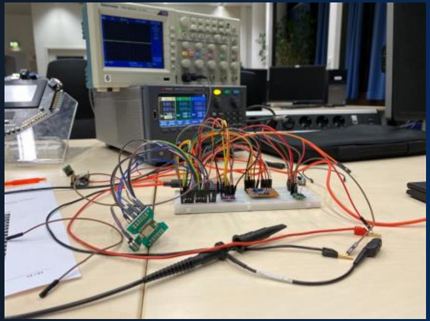
Gateway-Cloud Next Steps

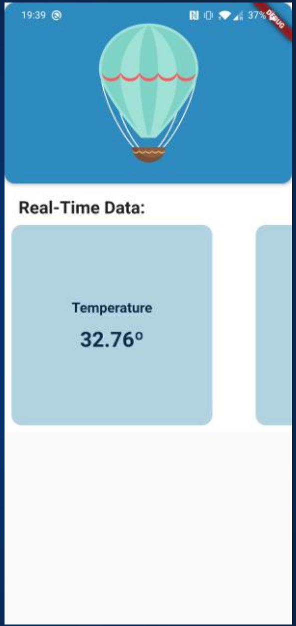
- Implement ThingsNetwork API for fetching data from there
- Design and implement cluster calculation for balloon clusters
  - Balloons will be configured in the gateway as to clusters
  - Temperature / humidity / pressure averages are calculated
  - If temperature of balloon xy in cluster is very high above the average, display warning (might be a fire)

App Showcase



Prototype Showcase

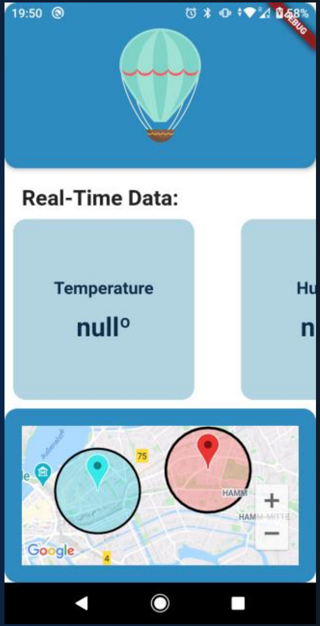




Real-Time Data:

Temperature

32.76°



Real-Time Data:

Temperature

null°

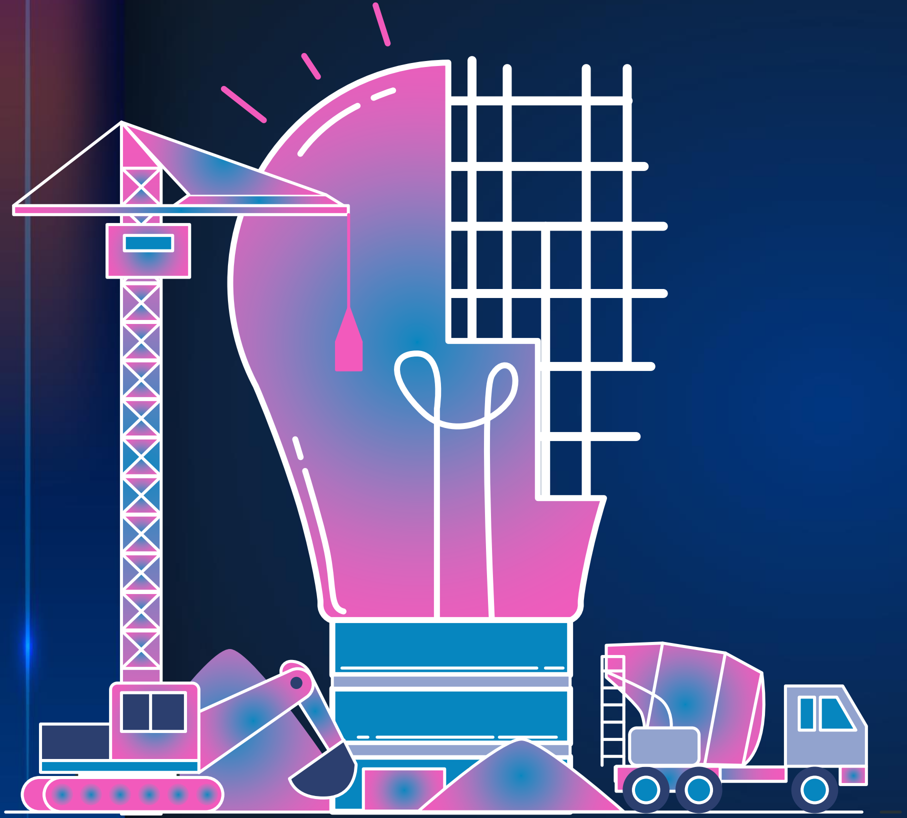
Hu

n

Next Steps

- Enable Communication with LoRa
- Measure Altitude using air pressure
- Restrict Graphic to a given time period
- Mount the sensors to the balloon

…more steps in the upcoming weeks



Thanks!

Do you have any questions?

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik.