foto: Io Cooman, Volkskrant

on-line slides: http://behouddeparel.nl/MySense%20status#januari2022

MySense: Sensor Kits
How Did They Do It

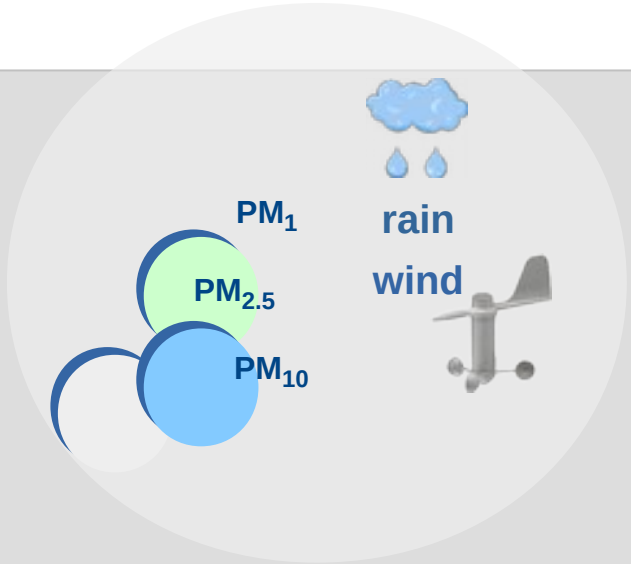
- how the measurement kit is done
  - dust and meteo sensoring in an agricultural region
  - (embedded) software fully Open Source
  - interface to data acquisition Python software
- Measurement Data Exchange Format (first implementation)

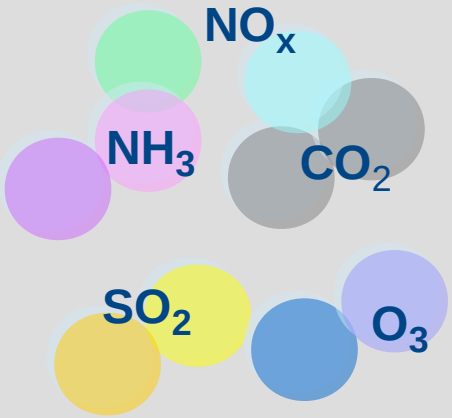
teus hagen
email: mysense@behouddeparel.nl

vereniging Behoud de Parel
http://behouddeparel.nl

Koppelting Meet je Stad 29-30 Januar 2022

a collection of measurement kits in Holland



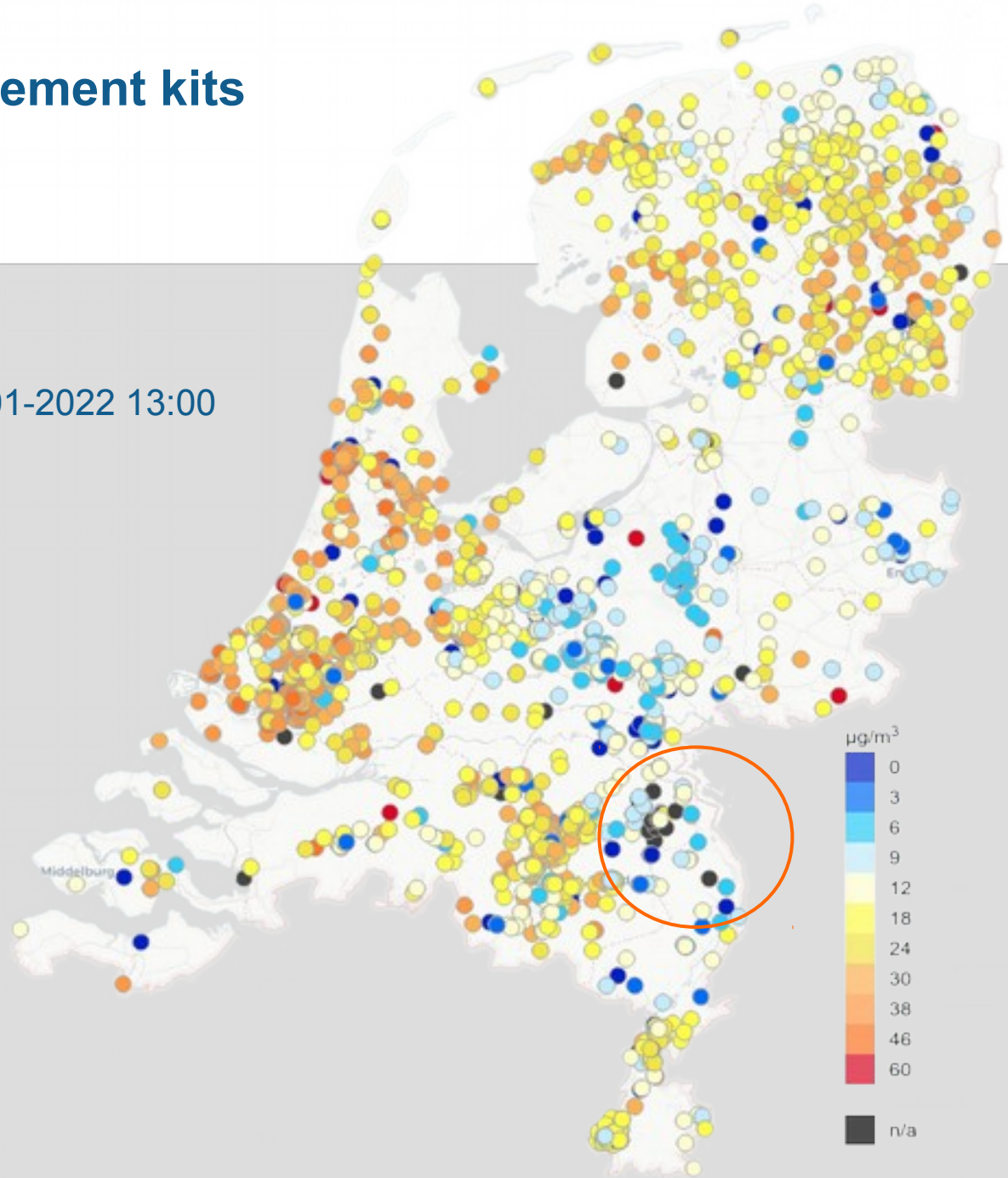


emissions:
nature
human
traffic
industry
farming

air quality citizen measurement kits

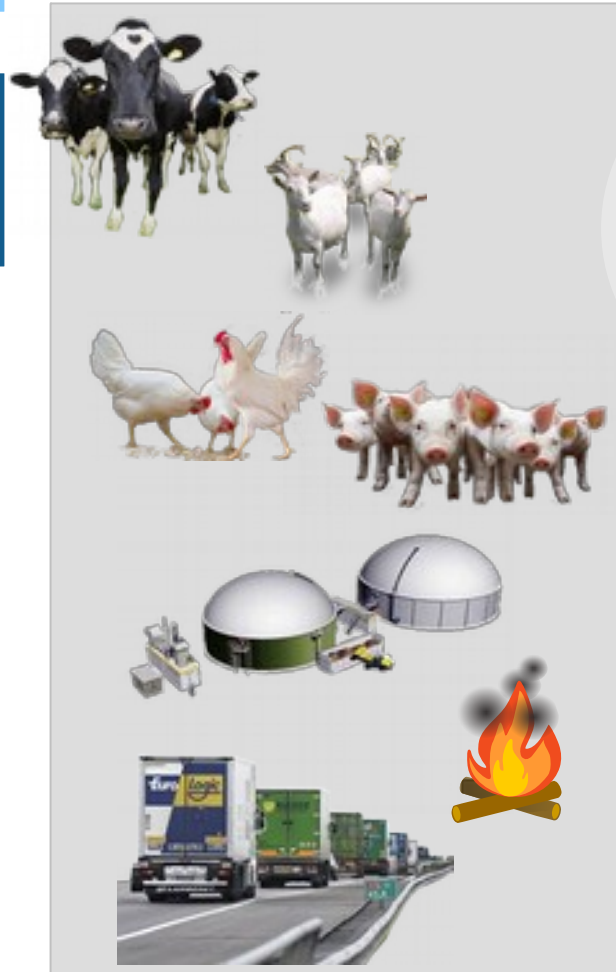
kits as known by RIVM

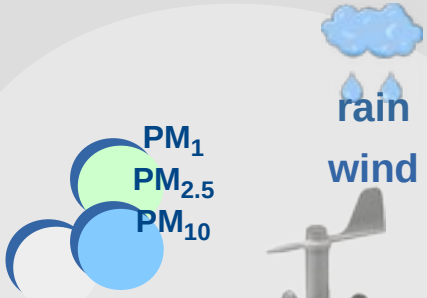
PM2.5 plausibility values at 24-01-2022 13:00

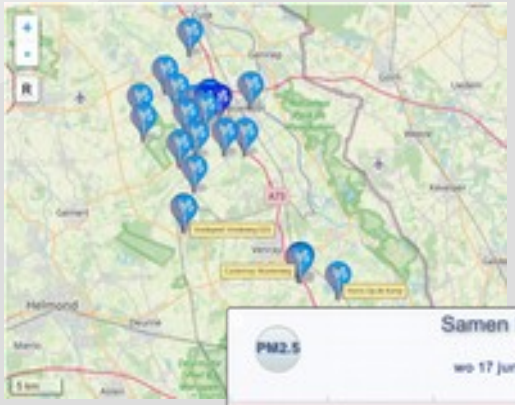


source: Samen Meten RIVM

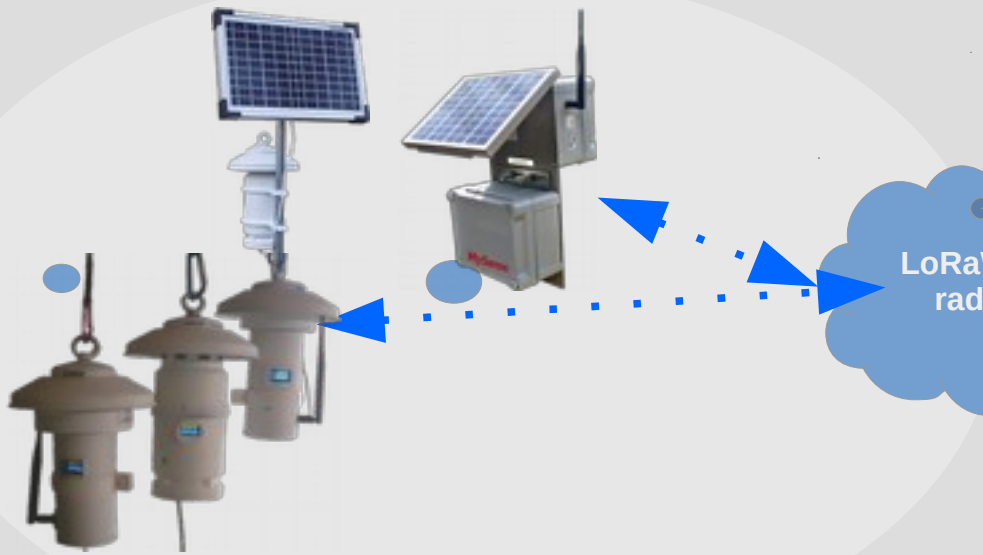
MySense measurement sensor kits







MySense architecture overview



TTN server

MySense
data collector

MyL database and forwarder

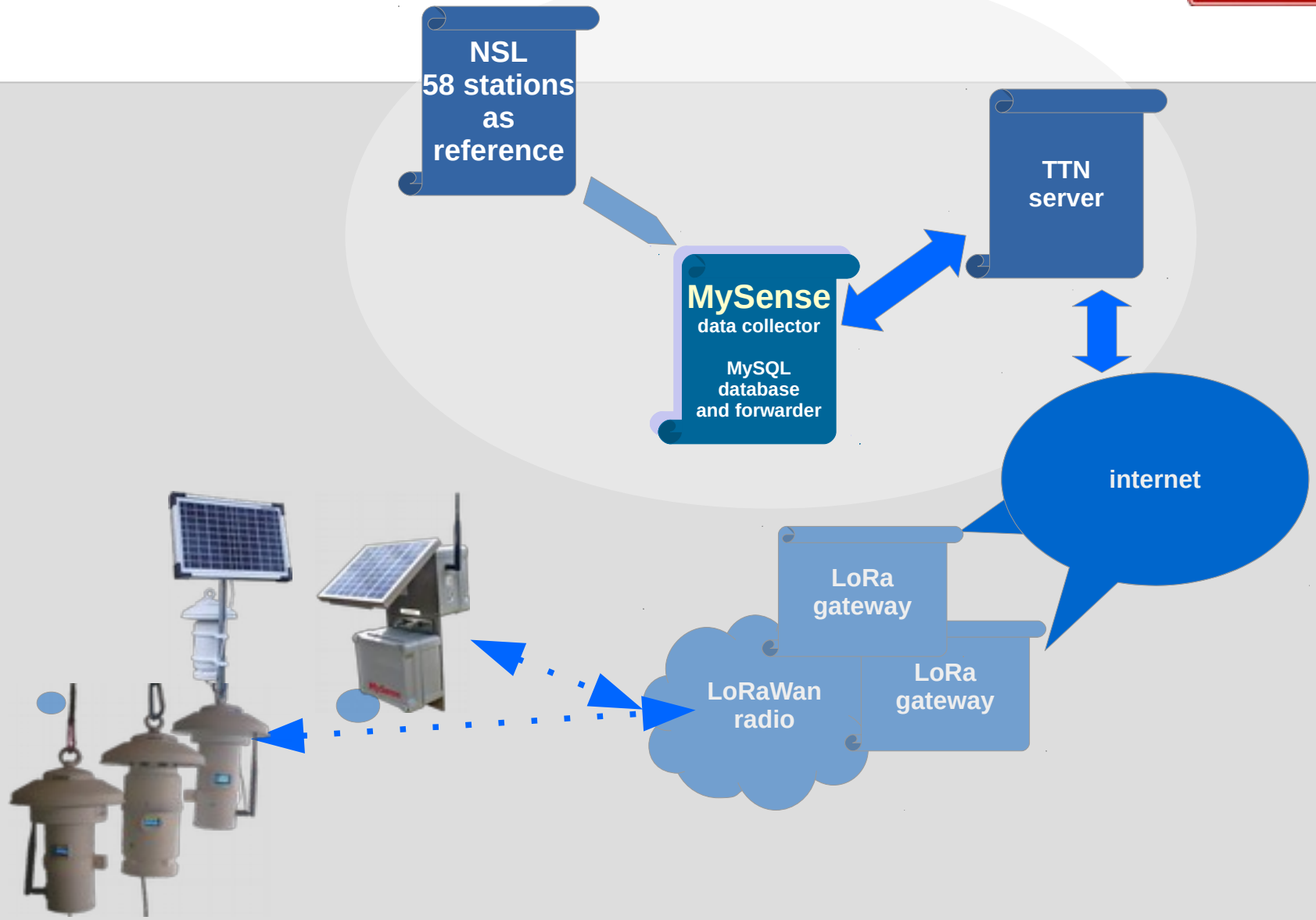
internet

LoRa gateway

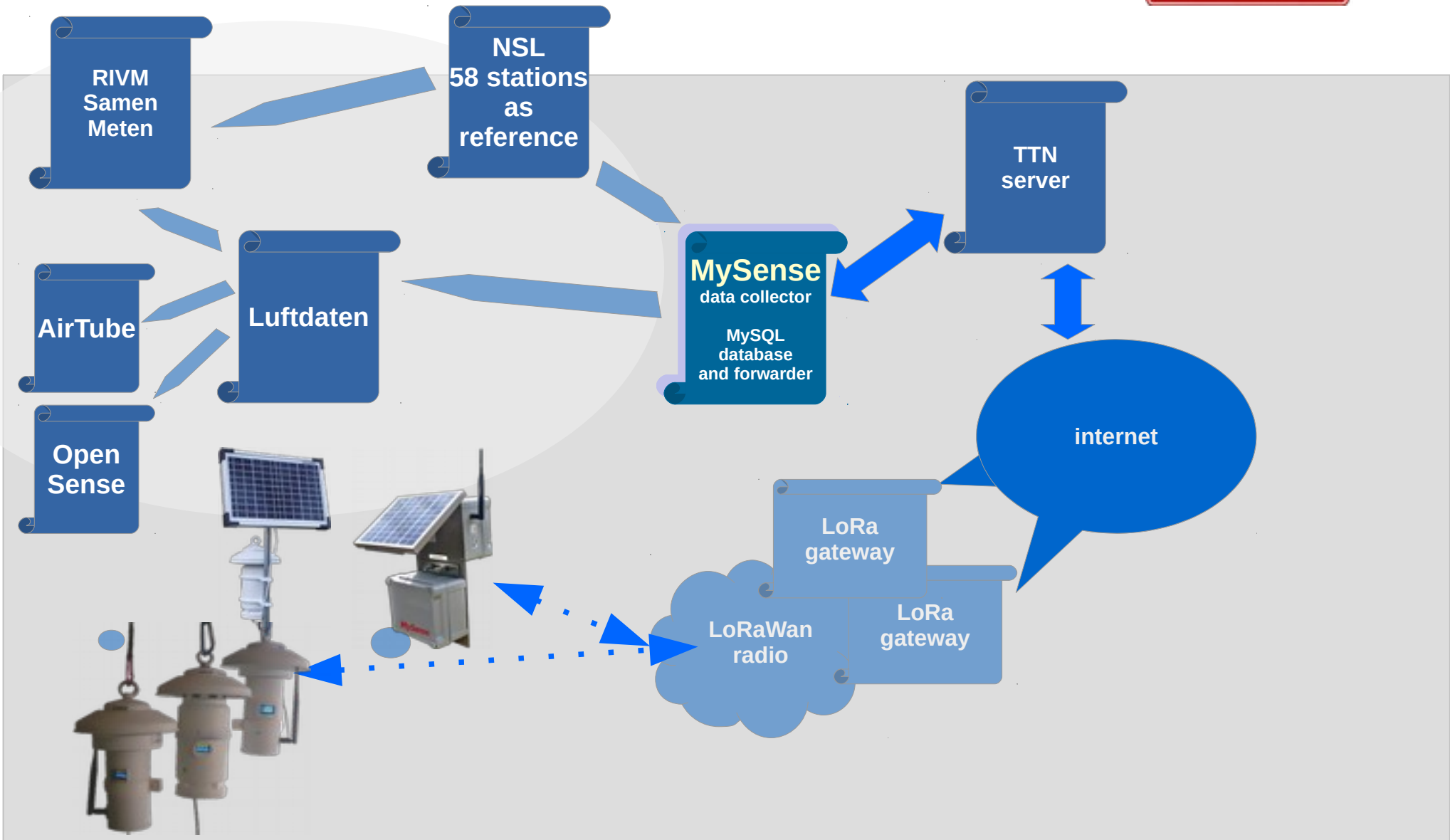
LoRa gateway

LoRaWan radio

MySense architecture overview



MySense architecture overview



MySense architecture overview

MySense challenge

http://behouddeparel.nl/MySense
https://github.com/teusH/MySense folder pycom

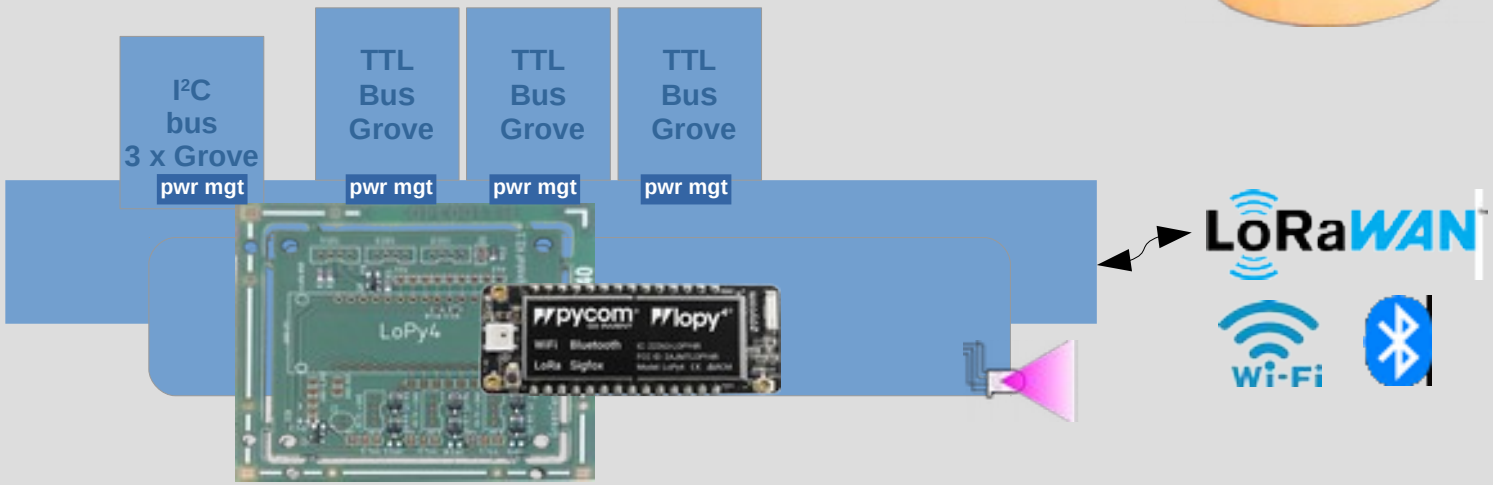
'Lego' is our inspiring source

modular, Open Source and high level modern programming (micro)Python

- **the easy (???) part is the hardware**:
  - choice of sensor (outdoor):
    dust sensor Sensirion (was Plantower),
    meteo Sensirion SHT31, was Bosch BME280 or 680 (VOC)
  - **sensor issues**: meteo Bosch BME and others fail in time: 50% fall out
  - V230 is a No-Go, so **solar and battery**: LiPo with special accu regulator and protection
    energy control is done in software via 'deep-sleep'
    flash memory has problems: limitation in non-volatyle memory use
  - ESP **microprocessor LoPy-4** PyCom with Open Source **microPython**
  - **modular plug and play**
    hardware I2C-bus, TTL via Processor Connector Board (PCB)
    software and use of multi threading
  - wifi is a No-Go: **LoRaWan** and so payload compression
    The Things Network and Mosquitto – JSON input TTN data records
  - **casing** simple and double DIY **PVC**

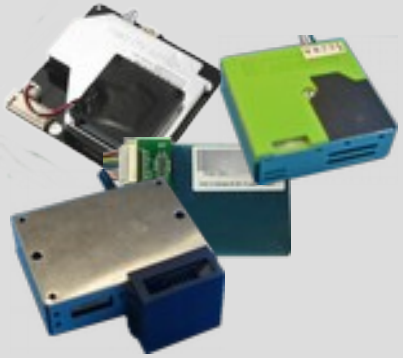
MySense sensor kit hardware

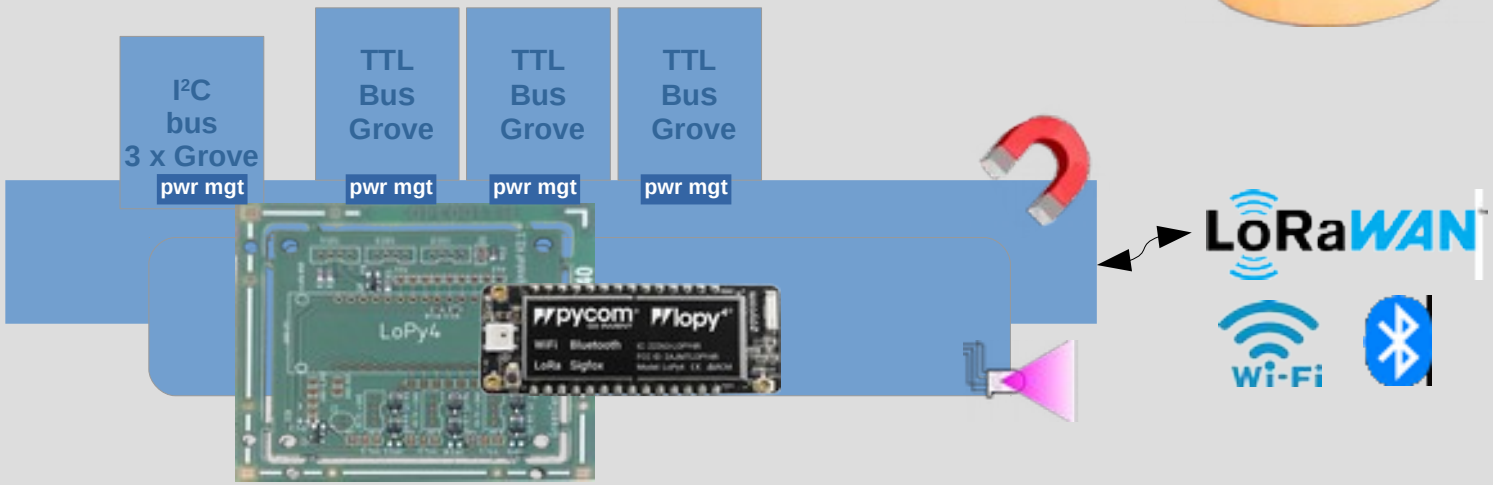
Processor Connector Board



MySense sensor kit hardware

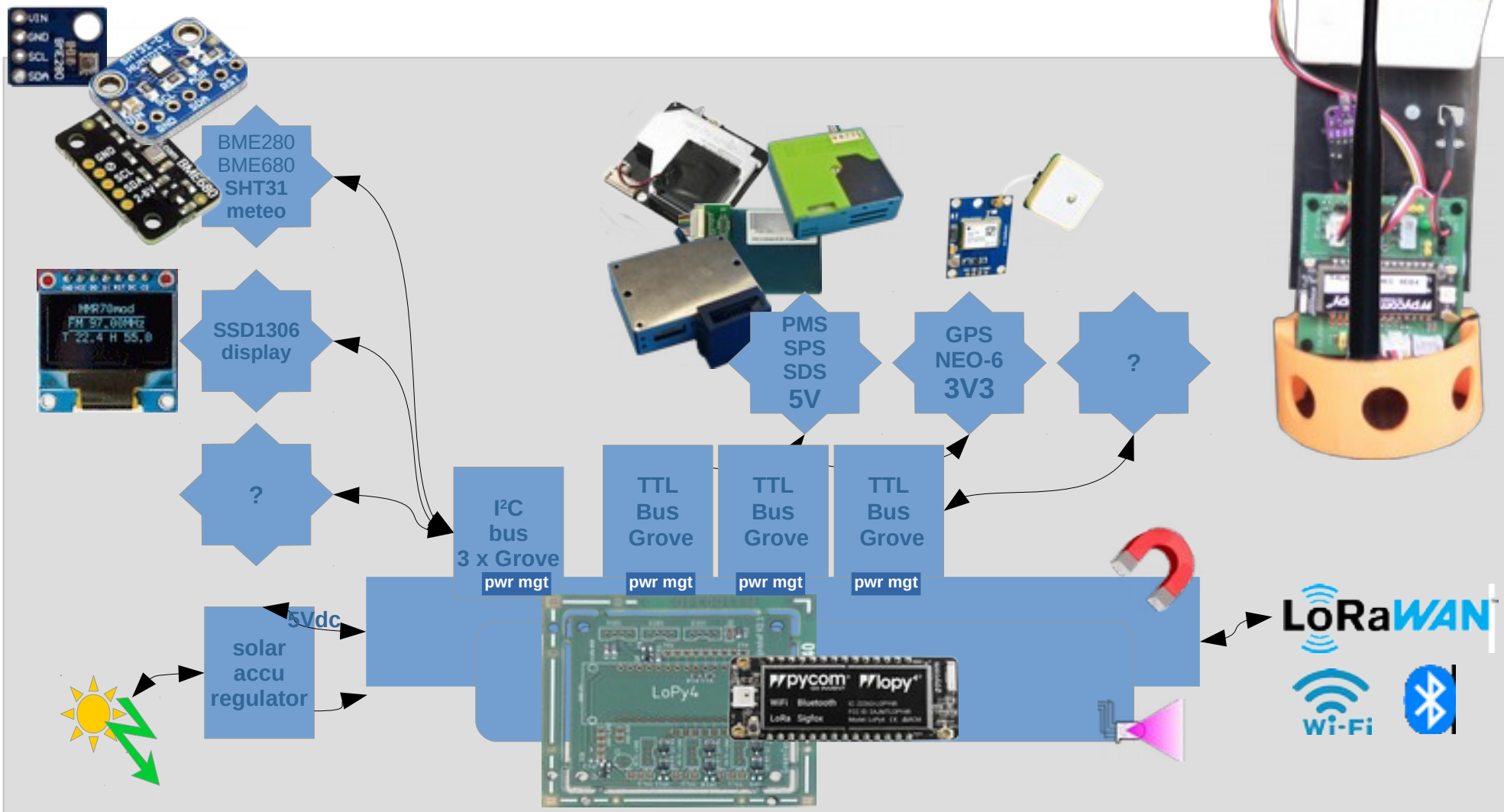
Processor Connector Board





MySense sensor kit hardware

Processor Connector Board

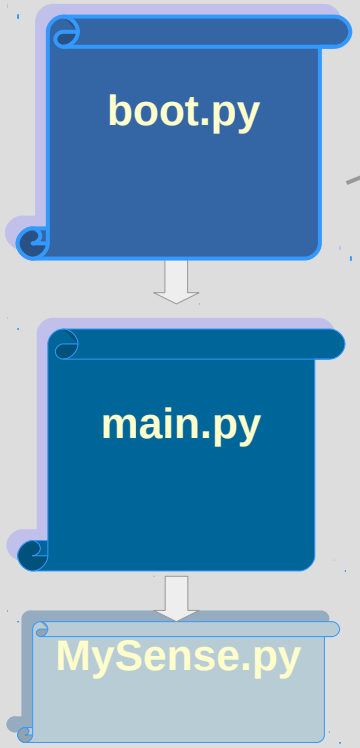


MySense sensor kit software

flash memory file system has all python scripts

using **Open Source** objective embedded Python 3

on PyCom Lopy-4 via wifi ftp/telnet or IDE atom



power on reason: wake up, power on

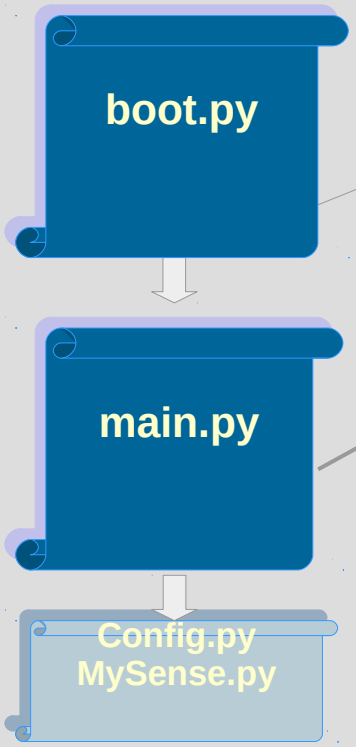
wifi AP config

MySense sensor kit software

flash memory file system has all python scripts

using **Open Source** objective embedded Python 3

on PyCom Lopy-4 via wifi ftp/telnet or IDE atom



power on reason: wake up, power on

wifi AP config

- Read-Eval-Print-Loop (REPL) modus
- wake up modus: power on, wake up, alarm
- check accu level / deep sleep again
- and run MySense loop ...

MySense main part

- central configuration
- sensing / send-receive loop

non volatyle SRAM:
alarm nr
LoRa status
loop count
accu statistics
GPS

on wakeup
config state
JSON file

lib modules:
device drivers
device detection

Config.py

MySense.py

power
actions
routines

init config

main loop

MySense main part

- central configuration
- sensing / send-receive loop

non volatyle SRAM:
alarm nr
LoRa status
loop count

on wakeup
config state
JSON file

lib modules:
device drivers
device detection

Config.py

- LoRaWan keys
- sample, idle rates
- pin configuration
- calibration/correction
- deepsleep modus
- device power on/off

MySense.py

power
actions
routines

init config

main loop

MySense main part

- central configuration
- sensing / send-receive loop

non volatyle SRAM:
alarm nr
LoRa status
loop count
accu statistics
GPS

on wakeup
config state
JSON file

lib modules:
device drivers

Config.py

- LoRaWan keys
- sample, idle rates
- pin configuration
- calibration/correction
- deepsleep modus
- device power on/off

MySense.py

power
actions
routines

init config

main loop

- power mgt devices
- LoRa payload compression
- wifi power mgt
- TTL / I2C configuration, device/sensor detection, driver binding
- network configuration
- initialisation net, display, meteo, dust, data, accu
- collect sensor data
- remote control (call back): power off, display/wifi on/off, rates

MySense main part

- central configuration
- sensing / send-receive loop

non volatyle SRAM:
alarm nr
LoRa status
loop count
accu statistics
GPS

on wakeup
config state
JSON file

lib modules:
device drivers
device detection

Config.py

- LoRaWan keys
- sample, idle rates
- pin configuration
- calibration/correction
- deepsleep modus
- device power on/off

MySense.py

power
actions
routines

init config

main loop

- power mgt devices
- LoRa payload compression
- wifi power mgt
- TTL / I2C configuration, device/sensor detection, driver binding
- network configuration
- initialisation net, display, meteo, dust, data, accu
- collect sensor data
- ren

- definition globals, accu statistics, event handling
- watchdog initialisation, display startup
- status and device detection
- power up modus

MySense main part

- central configuration
- sensing / send-receive loop

non volatyle SRAM:
alarm nr
LoRa status
loop count
accu statistics
GPS

on wakeup
config state
JSON file

lib modules:
device drivers
device detection

Config.py

- LoRaWan keys
- sample, idle rates
- pin configuration
- calibration/correction
- deepsleep modus
- device power on/off

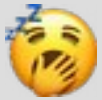
MySense.py

power
actions
routines

init config

- power mgt devices
- LoRa payload compression
- wifi power mgt
- TTL / I2C configuration, device/sensor detection, driver binding

main loop



- check accu level
- initialize devices/sensors, send meta data
- do measurements: dust, meteo, gps (distance)
- send data record, save LoRa state, call back request
- update watch dog
- display results, RGB led handling
- deepsleep or sleep and redo loop

MySense main part

- central configuration
- sensing / send-receive loop

non volatyle SRAM:
alarm nr
LoRa status
loop count
accu statistics
GPS

on wakeup
config state
JSON file

lib modules:
device drivers
device detection

Config.py

- LoRaWan keys
- sample, idle rates
- pin configuration
- calibration/correction
- deepsleep modus
- device power on/off

MySense.py

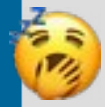
power actions routines

- power mgt devices
- LoRa payload compression
- wifi power mgt
- TTL / I2C configuration, device/sensor detection, driver binding
- network configuration
- initialisation net, display, meteo, dust, data, accu
- collect sensor data
- remote control (call back): power off, display/wifi on/off, rates

init config

- definition globals, accu statistics, event handling
- watchdog initialisation, display startup
- status and device detection
- power up modus

main loop



- check accu level
- initialize devices/sensors, send meta data
- do measurements: dust, meteo, gps (distance)
- send data record, save LoRa state, call back request
- update watch dog
- display results, RGB led handling
- deepsleep or sleep and redo loop

what is the kit saying to us?

- **The Thing Network** communication details
  - counter
  - gateway(s) statistics
  - TTN application id and TTN end node name
- measurement data in **payload format**:
  - format port 3: meta data periodically
  - format port 2/4/10/12: measurement data every ca 18 minutes

input measurement record details in Python

meta data part


```
{
  "timestamp": 1643071842,
  "id": { "project": "HadM", "serial": "e101e82a2c" },
  "net": {
    "type": "TTNV3",
    "gateways": [
      { "gateway_id": "rak7258-1", "rssi": -62, "snr": 9.8, "geohash": "u1hke8gfc6r" },
      { "gateway_id": "ic880a-pi",  "rssi": -40, "snr": 9.8, "geohash": "u1hke8gep3z" }
    ],
    "TTN_id":    "salk-20190518",
    "TTN_app": "201802215971az"
  },
  "meta": {
    "version":       0.5,
    "dust":          "SPS30",
    "gps":           "NEO-6",
    "meteo":         "BME680",
    "geolocation": { "geohash": "u1hke8gdzbr", "alt": 11.5 }
  }
}
```

input measurement record

in Python

meta data part


```
{
  "timestamp": 1643071842,
  "id": { "project": "HadM", "serial": "e101e82a2c" },
  "net": {
    "type": "TTNV3",
    "gateways": [
      { "gateway_id": "rak7258-1", "rssi": -62, "snr": 9.8, "geohash": "u1hke8gfc6r" },
      { "gateway_id": "ic880a-pi",  "rssi": -40, "snr": 9.8, "geohash": "u1hke8gep3z" }
    ],
    "TTN_id":    "salk-20190518",
    "TTN_app": "201802215971az"
  },
  "meta": {
    "version":       0.5,
    "dust":          "SPS30",
    "gps":           "NEO-6",
    "meteo":         "BME680",
    "geolocation": { "geohash": "u1hke8gdzbr", "alt": 11.5 }
  }
}
```

input measurement record

in Python

meta data part


```
{
  "timestamp": 1643071842,
  "id": {"project": "HadM", "serial": "e101e82a2c"},
  "net": {
    "type": "TTNV3",
    "gateways": [
      {"gateway_id": "rak7258-1", "rssi": -62, "snr": 9.8, "geohash": "u1hke8gfc6r"},
      {"gateway_id": "ic880a-pi",  "rssi": -40, "snr": 9.8, "geohash": "u1hke8gep3z"}
    ],
    "TTN_id":    "salk-20190518",
    "TTN_app": "201802215971az"
  },
  "meta": {
    "version":       0.5,
    "dust":          "SPS30",
    "gps":           "NEO-6",
    "meteo":         "BME680",
    "geolocation": {"geohash": "u1hke8gdzbr", "alt": 11.5 }
  }
}
```

input measurement record

in Python

measurements data part


```
{
  "timestamp": 1643119378,
  "id":        { "project": "SAN", "serial": "b4e62df5571d" },
  "net":       {
    "type":        "TTNV3",
    "gateways": { "gateway_id": "a57", "rssi": -83, "snr": 11, "geohash": "u1hjeu8qq" },
    "TTN_id":     "bwlvc-571d",     "TTN_app": "201802215971az"
  },
  "data":       {
    "version":  1.8,
    "SPS30":     {
      "pm05_cnt": 5501.1,   "pm1_cnt":   7809.6, "pm4_cnt":   12334.8,
      "pm4_cnt":   12334.8, "pm25_cnt": 8919.1, "pm10_cnt": 12358.8,
      "pm1":         40.0,      "pm25":        51.4,     "pm10":       61.6,
      "grain": 0.4
    },
    "SHT31":    [ { "temp": 6.0, "rv": 71.4 },   { "temp": 6.7, "rv": None } ],
    "NEO-6":    {  "geohash": "u1hjeucfsey", "alt": 33.0 }
  }
}
```

input measurement record

in Python

measurements data part


```
{
  "timestamp": 1643119378,
   "id":          { "project": "SAN", "serial": "b4e62df5571d" },
   "net":         {
    "type":       "TTNV3",
    "gateways": { "gateway_id": "a67", "rssi": -83, "snr": 11, "geohash": "u1hjeu8qq" },
    "TTN_id":     "bwlvc-571d",     "TTN_app": "201802215971az"
   },
   "data":        {
    "version":  1.8,
    "SPS30":   {
      "pm05_cnt": 5501.1,   "pm1_cnt":   7809.6, "pm4_cnt":   12334.8,
      "pm4_cnt":   12334.8, "pm25_cnt": 8919.1, "pm10_cnt": 12358.8,
      "pm1":         40.0,      "pm25":        51.4,     "pm10":       61.6,
      "grain": 0.4
    },
    "SHT31":   [ { "temp": 6.0, "rv": 71.4 },   { "temp": 6.7, "rv": None } ],
    "NEO-6":   {  "geohash": "u1hjeucfsey", "alt": 33.0 }
   }
}
```

questions, comments

***thank you for your support*:**

Fontys Venlo GTL, MilieuDefensie, Kipster, Frank Koenders e.a., municipality St. Anthonis, foundation Burgerwetenschappers Land van Cuijk, association Behoud de Parel, RIVM, daily newspapers Trouw and de Volkskrant, local papers de Gelderlander and de Limburger.