Open-source data logger for environmental data collection

City Regions and Businesses boosting Nature Positivity (KYLKI)

Keijo Länsikunnas

AmiEs-2026

Metropolia
University of Applied Sciences

U!REKA
EUROPEAN UNIVERSITY

KYLKI

Kaupunki ja Yritykset Luontopositiivisuuden Klrittäjinä

- City Regions and Businesses boosting Nature Positivity (KYLKI) is a collaborative project of the city of Vantaa, Metropolia UAS, Laurea UAS, and the Finnish Environment Institute.
- The project members are working together to support the transition of businesses and cities in the metropolitan area towards nature-positive practices.
- The project aims to promote solutions that foster sustainable business, environmental responsibility, and help cities achieve their biodiversity goals.
- Through activities such as data collection, co-creation within an RDI framework, and pilot projects, the project will create opportunities for business innovation, strengthen the competitiveness of small and medium-sized enterprises (SMEs), and promote the development of environmentally responsible markets.

About environmental sensors

- Sensors for environmental measurements have varying interfaces
  - SDI-12
  - Modbus
  - BLE
  - Etc.
- Data loggers are available
  - Very often sensor manufacturers also sell data loggers
  - Sensors with standard interface – especially SDI-12 are often compatible across vendors
    - Configuration is required for cross vendor compatibility

The problem

- Connected data loggers are typically vendor locked – data can be stored only in the vendor's cloud infrastructure and requires a subscription
- Sensor protocol support is limited – compatible sensors can be added through configuration but there is no way add new communication protocols or new interfaces
  - Configuration is limited to reading and labeling data read from sensors using existing protocols
  - Some loggers support data processing with a scripting language – scripts can only do some math on the received data but can't be used to implement new protocols
- Only a single data endpoint (in cloud) is supported

Flexible and extendable open-source data logger

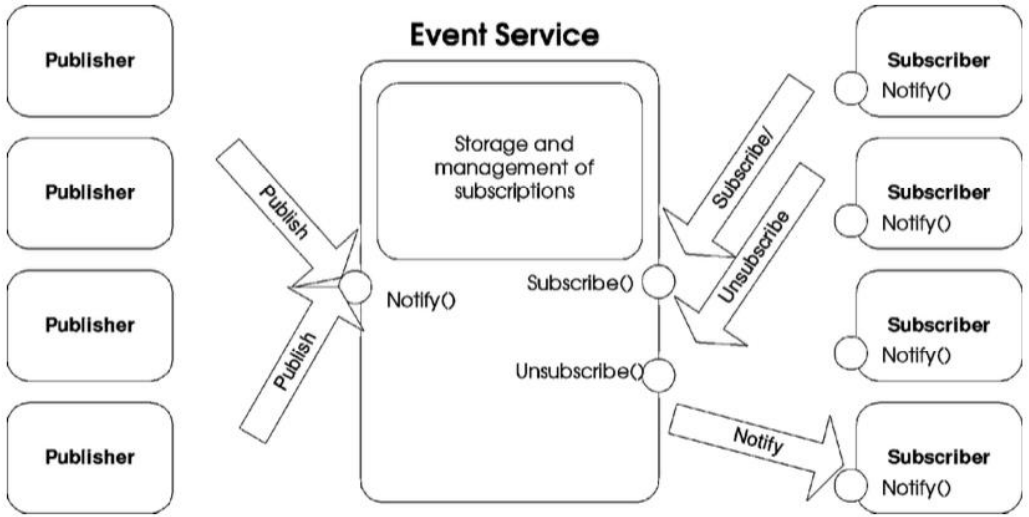
Support for multiple (simultaneous) cloud end points

Fully programmable in C++

New interfaces/protocols can be added easily

Open-source data logger

- Publish/subcribe interaction scheme is widely used in IoT applications
- Publish/subscribe paradigm decouples communication entities in time, space, and synchronization
- By applying the paradigm throughout the whole system changes in system can be implemented independently of each other

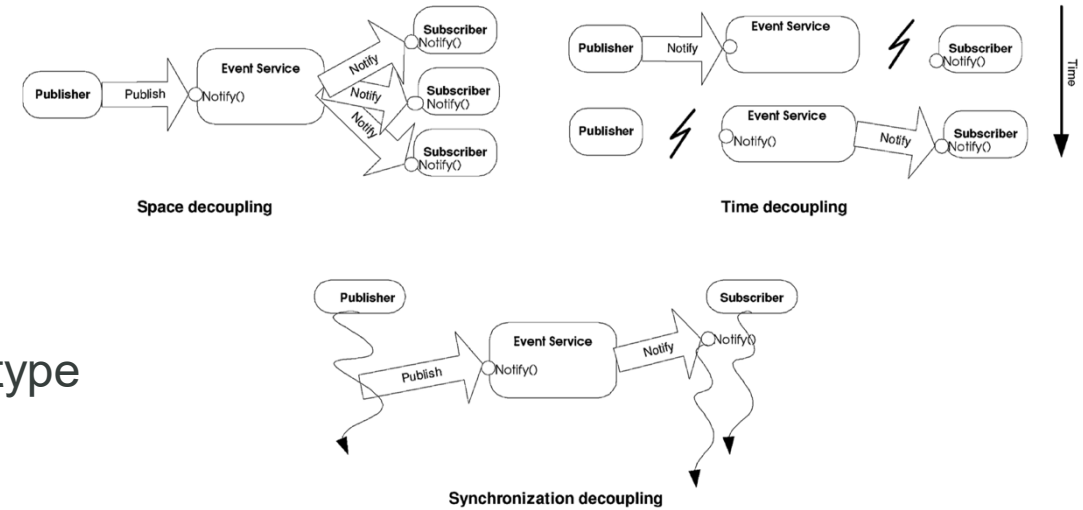


The Many Faces of Publish/Subscribe ACM Computing Surveys, Vol. 35, No. 2, June 2003, pp. 114–131.

Pub/sub

- Data logger has three main components
  - Publisher/subscriber mechanism
    - Topics – classes to hold data from one sensor type
    - You can publish/subscribe to a topic
    - Typically there is a dedicated thread for each sensor type
  - Transport – implements connection to cloud
  - Serializer – converts topic data to a suitable format (eg. JSON)
- Cloud sender thread
  - Wakes up at configurable intervals – interval is not dependent on sensor reading frequency
    - Requires queueing system to hold data while sender is inactive
  - Uses serializer and transport

The Many Faces of Publish/Subscribe ACM Computing Surveys, Vol. 35, No. 2, June 2003, pp. 114–131.

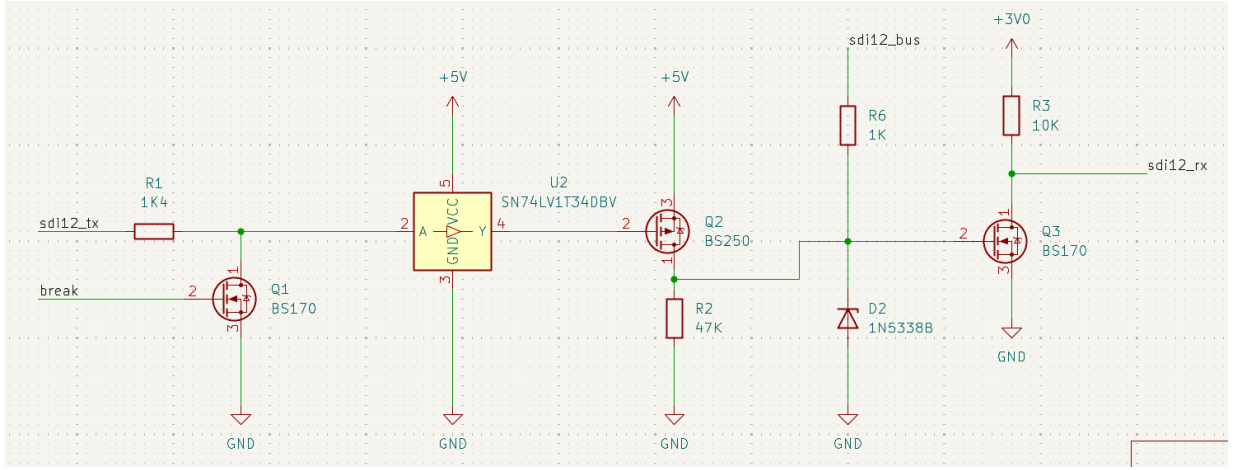


SDI-12 scanner

- SDI-12 is UART based protocol that runs at 1200 bps
- nRF UART is somewhat limited – there is no support for generating break condition
  - Requires extra GPIO pin to generate break
- SDI-12 publisher scans up to 10 devices on the bus
  - Detects type of device and registers recognized types (Teros10, Teros12, Solyx14, Solinst Levelogger) as publishers

SDI-12 interface

- SDI-12 is UART based protocol that runs at 1200 bps
- Signaling similar to RS-232 except there is no negative voltage (mark = 0V, space = +5V)
- Surge/overvoltage protection needs to be implemented
- Multidrop bus
- Simplex communication



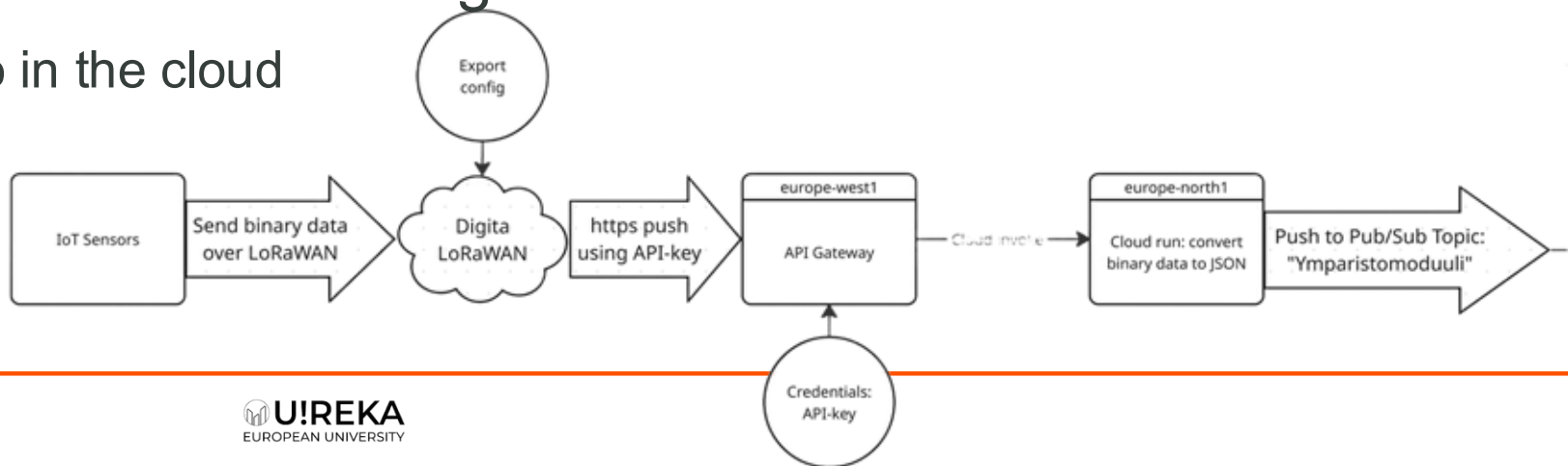
Ruuvi scanner

- Ruuvi Tags use BLE to communicate data
  - Either beacon mode or connection oriented
- Ruuvi scanner scans for beacon signals
  - Sensors to listen are configured in NV-storage
  - Unlisted beacons are ignored
  - Listed sensors are registered as publishers
- Configurable scanning interval



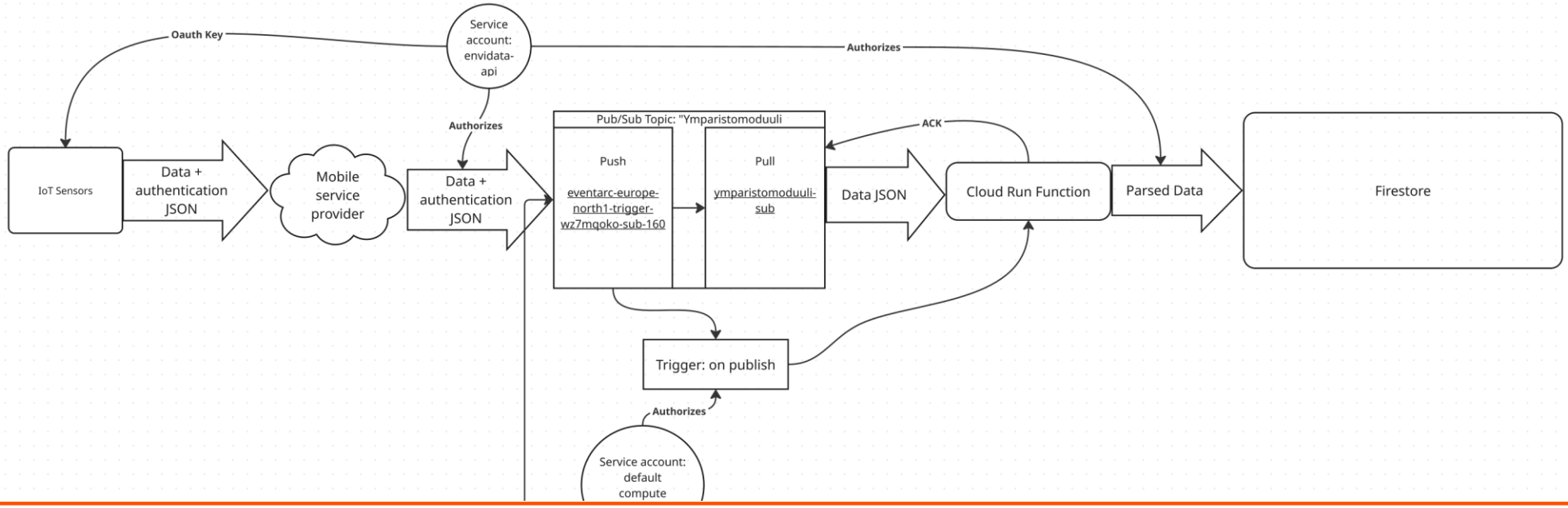
LoRaWAN transport

- Implements a LoRaWAN connection using AT-command based LoRaWAN module
  - Low-power mode to extend battery life. Synchronizes system clock to network time
  - Protocol buffers or proprietary encoding
- LoRaWAN credentials are stored in non-volatile storage and can be set using command shell during deployment
- API Gateway is used to publish data to Google cloud
  - Pub/sub paradigm is used also in the cloud

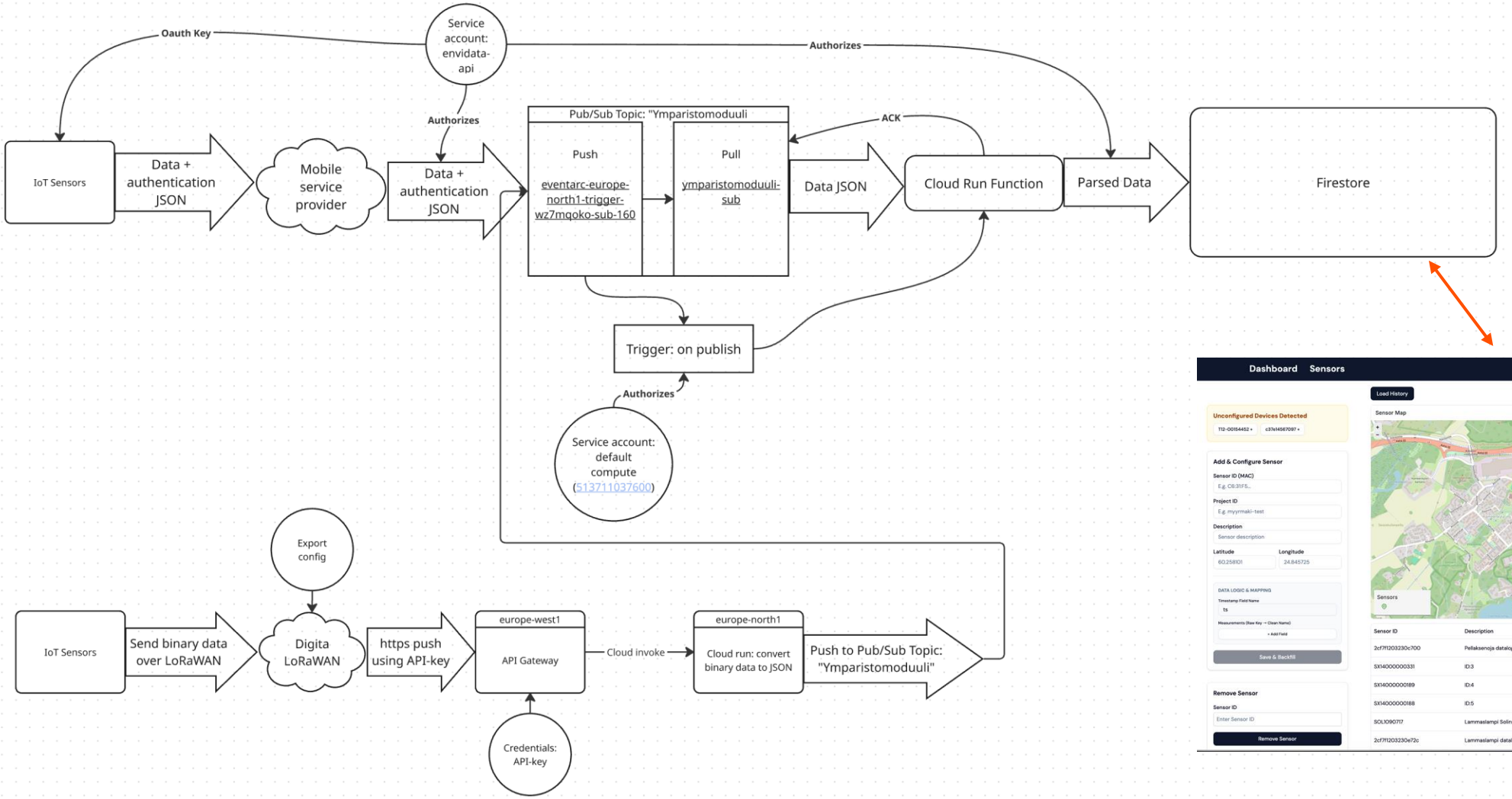


Modem transport

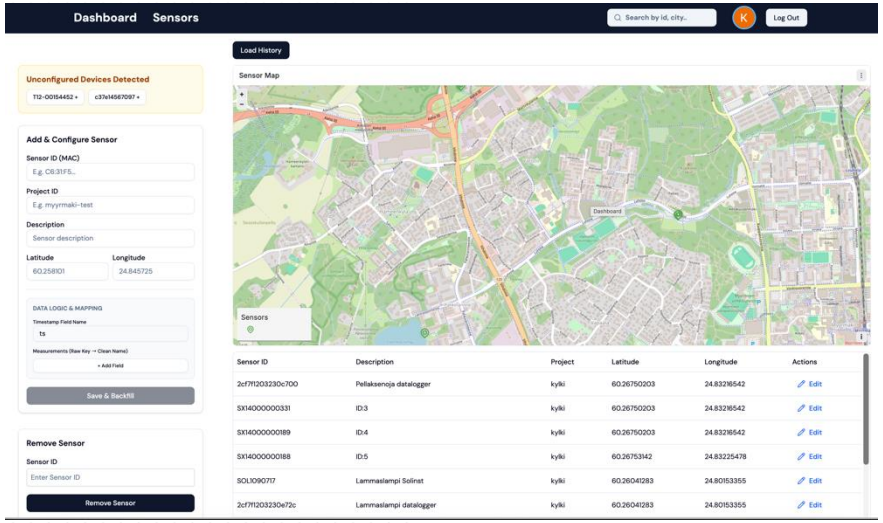
- Authenticates directly to Google pub/sub
- Data is sent in JSON so it can be stored in Firestore without conversion
  - 4G/5G has higher data bandwith so human readable data can be sent
  - Log entries are easier to read



The big picture



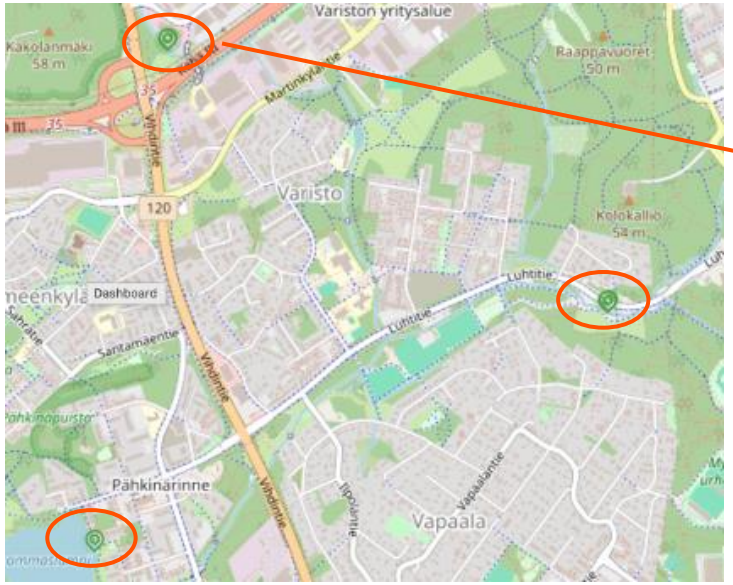
WEB UI runs in VM

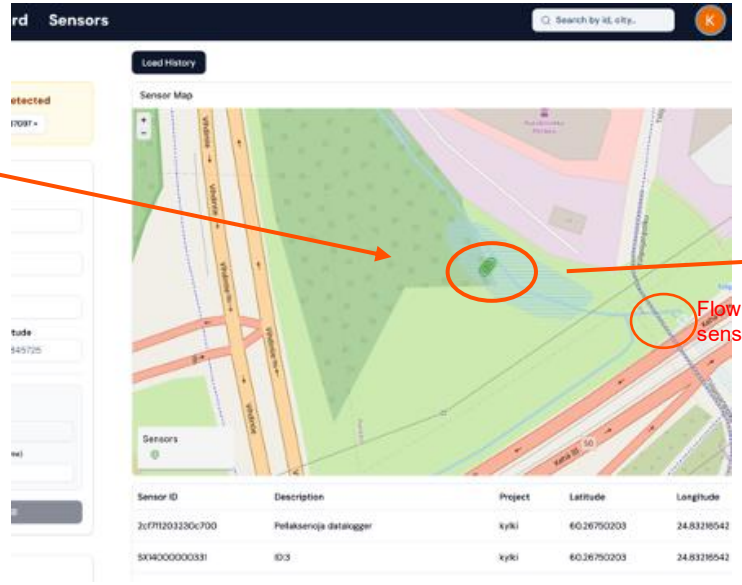


Current platform

- nRF52840 dongle
  - SDI-12 interface
  - BLE
  - 4G modem or LoRaWAN – can’t have both if SDI-12 is used because board has only two UARTs
- Runs Zephyr OS
  - Zephyr is Linux-like in the sense that it uses device tree which makes it easy switch to different hardware
  - Zephyr has a large ecosystem of drivers

Use case of data logger







City of Vantaa needed data for modeling functionality of rain gardens and storm water drainage in Varisto area. Two rain gardens and a pond were monitored using the data logger.

Three Solyx14 sensors were installed in each rain garden. Solinst Levelogger and a barometric pressure sensor were installed in Lammaslampi pond.

Flow rate in storm water drainage was monitored with a sensor that was not connected to the data logger. Measurements were taken at 10 minute intervals.

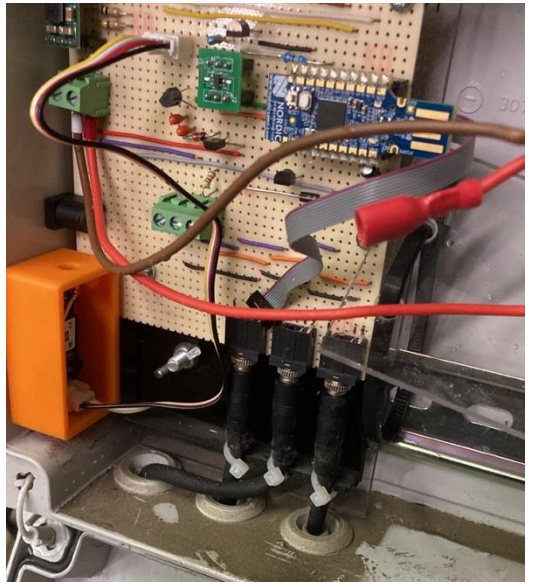
Sensor installation



Solyx14 sensors were installed underground and cables were run to data logger.

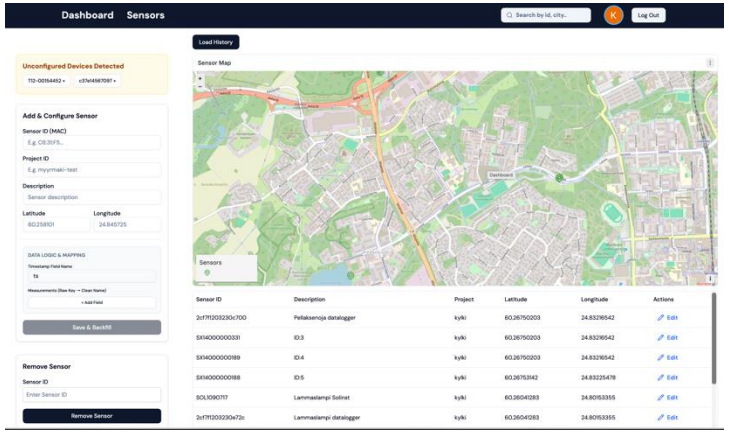


Solinst Levelogger was installed under water and the reference levels were taken by City of Vantaa land surveyors.

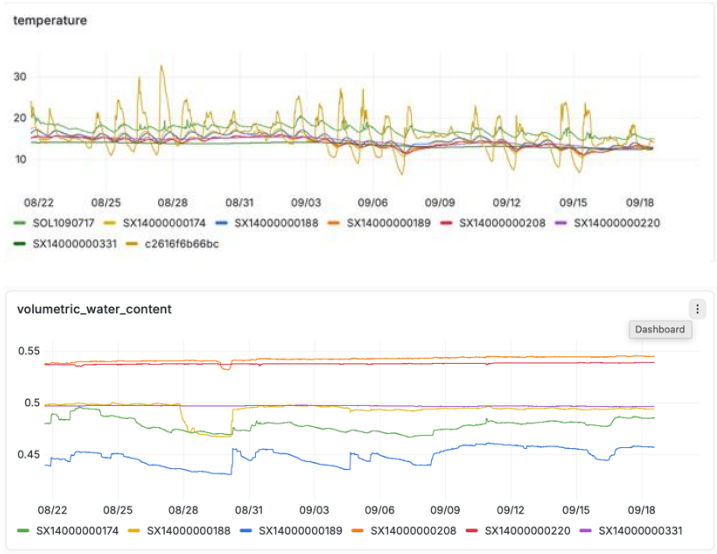


Inside of the data logger with battery removed. This variant was equipped with LoRaWAN tranceiver (orange box on the left).

Data storage



Collected data was stored in Firestore in Google cloud. A web client was implemented for easy inspection and export of the data.



Grafana was used for plotting trends and comparison graphs of the collected data.

