Senso

ontbossing detecteren met geluidsanalyse

Andrei Florian (Ierland)

Illegale houtkap is in veel landen een probleem. Het Senso-project biedt een mogelijke oplossing. Dit apparaat, gebaseerd op de Arduino MKR Fox, waarschuwt de autoriteiten wanneer het geluid van houtkap wordt gedetecteerd.

Toen ik in Roemenië in de bergen wilde gaan wandelen, viel het me op dat veel mensen zich zorgen maken over de illegale houtkap, en wilde ik meehelpen daar iets aan te doen. Ik heb het probleem bestudeerd en kwam tot de conclusie dat het mogelijk moest zijn om het geluid van houthakkersgereedschap te detecteren. Mijn oplossing wordt in dit artikel beschreven.

Het geluid van ontbossing

Ontbossing is één van de ernstigste problemen voor onze generatie. Overal in de wereld worden bossen gekapt om ruimte te maken voor landbouw en huizen. Soms worden daarvoor stukken bos afgebrand; en illegale houtkap is een internationaal verschijnsel. Ontbossing hangt rechtstreeks samen met klimaatverandering, want bij het verbranden van organisch materiaal komt veel kooldioxide vrij.

Houtkap is een groot probleem, zowel in ontwikkelde landen als in ontwikkelingslanden. Roemenië is een goed voorbeeld. Hoewel er regels zijn voor de houtkap in Roemenië, wordt er veel illegaal gekapt. Roemenië bezit 65% van het Europese oerbos, maar dat verdwijnt snel omdat het gekapt wordt.

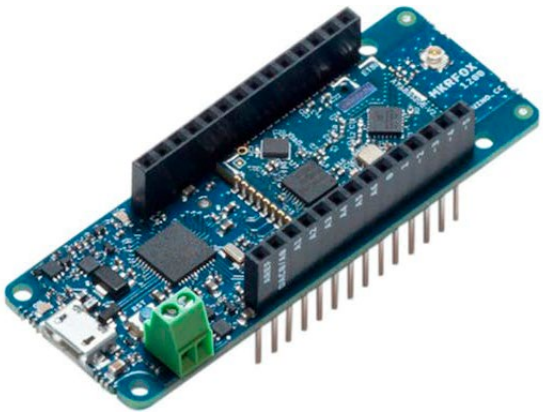
Meestal worden daar kettingzagen voor gebruikt, wat een hoop lawaai produceert. En dat is waar Senso kan helpen: een energiezuinig apparaat met een module die verschillende frequentiebanden kan onderscheiden. Het apparaat is geprogrammeerd om het geluid van houtkapmachines en -gereedschappen, zoals kettingzagen, te detecteren en de autoriteiten te waarschuwen. Elke vijftien minuten wordt een meting gedaan. Als een verdacht geluid wordt gedetecteerd, stuurt het apparaat een event naar het backend, waar de bosgebieden worden weergegeven op een landkaart.

Senso bestrijdt de illegale houtkap door meteen de autoriteiten te alarmeren als er houtkap wordt gedetecteerd, zodat die sneller in actie kunnen komen dan ooit en zich kunnen concentreren op de gebieden met de meeste houtkap.

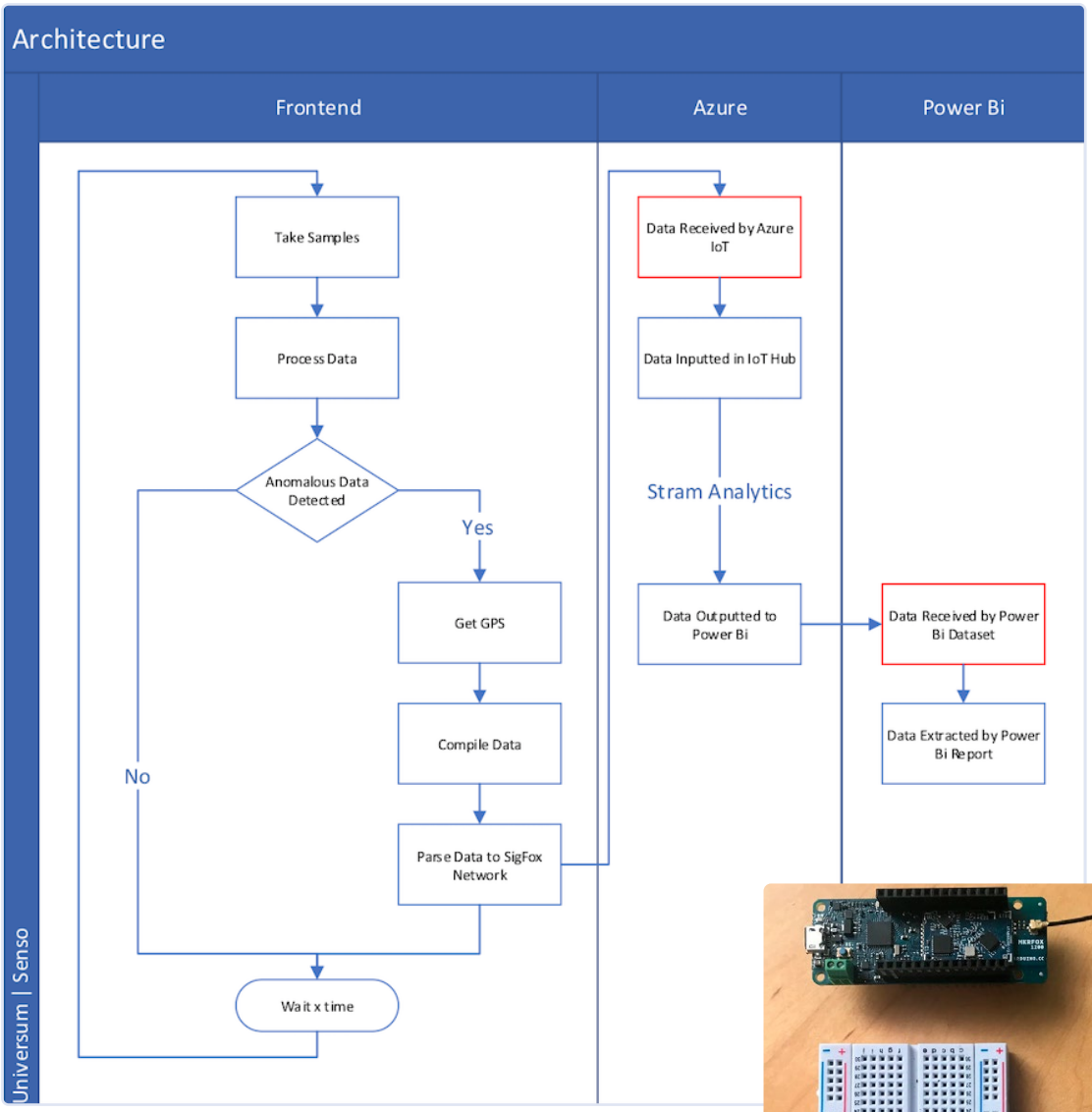
Overzicht

Ik heb voor dit project gekozen voor de Arduino MKR Fox (**figuur 1**) vanwege zijn energiezuinigheid. Ik heb hem al eerder toegepast in verschillende low-power communicatieprojecten. Het is op de MKR Fox heel gemakkelijk om Sigfox te gebruiken, want dat is ingebouwd. En dankzij Arduino-bibliotheken kunnen we er snel mee aan de slag.

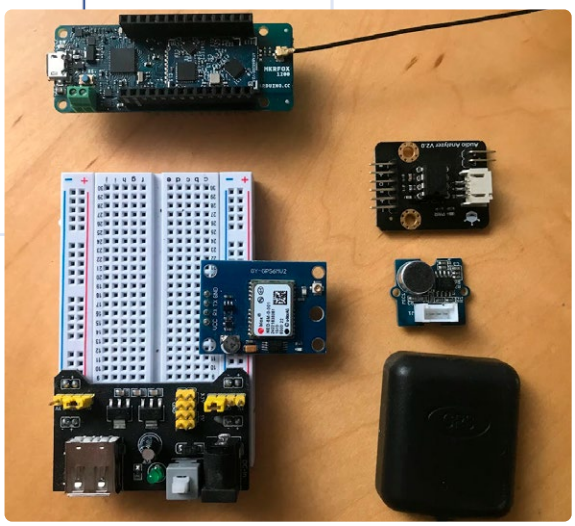
Ik heb besloten het apparaat met 5 V te voeden via een breadboard-voeding, omdat de modules te veel vermogen gebruiken om ze te voeden vanuit de processorkaart. Het gaat hier om een prototype; de capaciteit van de batterij is te klein om het geheel in het veld te gebruiken. Ik ben nog bezig met het uitschakelen van de sensoren als het apparaat in slaapstand staat om energie te sparen.



Figuur 1. De Arduino MKR Fox is het hart van dit project (bron: Arduino).



Figuur 2. De systeemarchitectuur als stroomdiagram.



Figuur 3. Deze componenten worden gebruikt in dit project.

Architectuur

Het project bestaat uit een frontend en een backend. Het frontend is het apparaat in het bos, dat geluidssamples neemt en het backend, dat de data verwerkt en weergeeft, bestaat uit Sigfox, Microsoft Azure en Microsoft Power BI, zie **figuur 2**.

Frontend: het apparaat ontwaakt met regelmatige tussenpozen om te luisteren naar geluid. Het zoekt naar activiteit in specifieke frequentiebanden die kenmerkend zijn voor kettingzagen en soortgelijke gereedschappen. Als zo'n geluid wordt gevonden, stuurt het apparaat zijn locatie en batterijniveau naar de cloud. Daarna gaat het weer slapen, totdat het tijd is om weer wakker te worden.

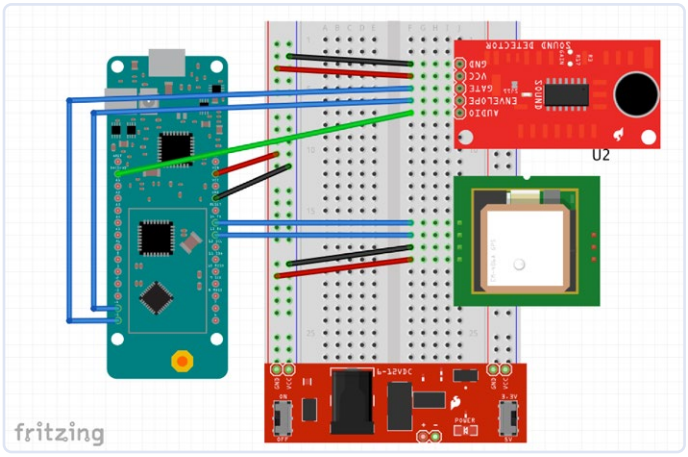
Een interessante uitbreiding zou een positie-tracker kunnen zijn. Met accelerometers zou het apparaat kunnen detecteren wanneer de boom waar het op is gemonteerd wordt neergehaald. Als dat gebeurt, zou het zijn locatie elke 10 minuten naar het backend kunnen sturen, zodat de autoriteiten de omgehakte boom tijdens het transport kunnen volgen.

Backend: het door de sensor gedetecteerde event wordt ontvangen door het Sigfox-backend, dat het doorstuurt naar Azure IoT met behulp van een callback-functie. Azure IoT wijst de data dan toe aan een hub. Een data-streamingdienst haalt de data op en zet die in een Power BI-dataset. Power BI haalt dan de gegevens uit de dataset om ze weer te geven in een rapport.

Daarna zou een berichtentoepassing kunnen worden aangeroepen om de autoriteiten te alarmeren als er een event is geregistreerd. Dan zouden boswachters precies weten waar ze naartoe moeten gaan als er een boom wordt gekapt.

De opbouw

Genoeg woorden – nu gaan we bouwen. Het project bestaat uit heel wat onderdelen (**figuur 3**): een Arduino MKR Fox-boardt, een GPS-module plus antenne, een GSM-antenne, een Audio Analyzer Module van DFRobot (zie figuur 3) en een microfoon. De onderdelen zijn op een breadboard aangesloten (**figuur 4**). De schakeling wordt via een 5V/3,3V-breadboard-voedingsmodule gevoed uit een 9V-batterij. De microcontroller wordt gevoed door de breadboard-voeding.



Figuur 4. Bedrading van het Senso-prototype.

Ik heb een goedkope GPS-module gebruikt die voor € 10 online te koop is. Hij is gemakkelijk te gebruiken, maar nogal traag met het vinden van de satellieten. Een goede antenne is wel noodzakelijk.
De Audio Analyser Module meet het geluidsniveau in zeven frequentiebanden: 63 Hz, 160 Hz, 400 Hz, 1 kHz, 2,5 kHz, 6,25 kHz en 16 kHz. Uitgebreid testen heeft me geleerd dat kettingzaaggeluiden in het bereik van 2,5 kHz tot 6,25 kHz vallen. Bij het afspelen van opgenomen kettingzaaggeluiden vertoonde de 6,25kHz-band pieken. Ik heb een rekenmethode ontwikkeld die voorkomt dat andere geluiden, zoals auto's en natuurgeluiden, herkend worden als een kettingzaag. Die methode is niet perfect, maar werkt toch redelijk betrouwbaar.
De code bestaat uit drie delen, die we nu zullen bespreken.

Geluid bemonsteren en verwerken

Dit deel van het programma wordt elke keer dat het apparaat ontwaakt uitgevoerd. Eerst worden samples van de geluidssensor genomen en dan wordt de ontvangen data verwerkt door de gezochte frequentieband te vergelijken met de andere banden. Dat gaat met de volgende regels code:

```
long comparison = ((valueMean[0] + valueMean[1] + value-
Mean[2] + valueMean[3]) / 1.9);
if (valueMean[5] > comparison) ...
```

De gemiddelde waarden van alle frequentiebanden (behalve de doelband) worden opgeteld en gedeeld door 1,9. Als de waarde van de doelband groter is dan `comparison`, wordt een positieve detectie verondersteld (**figuur 5**).

GPS-positie en batterijniveau verkrijgen

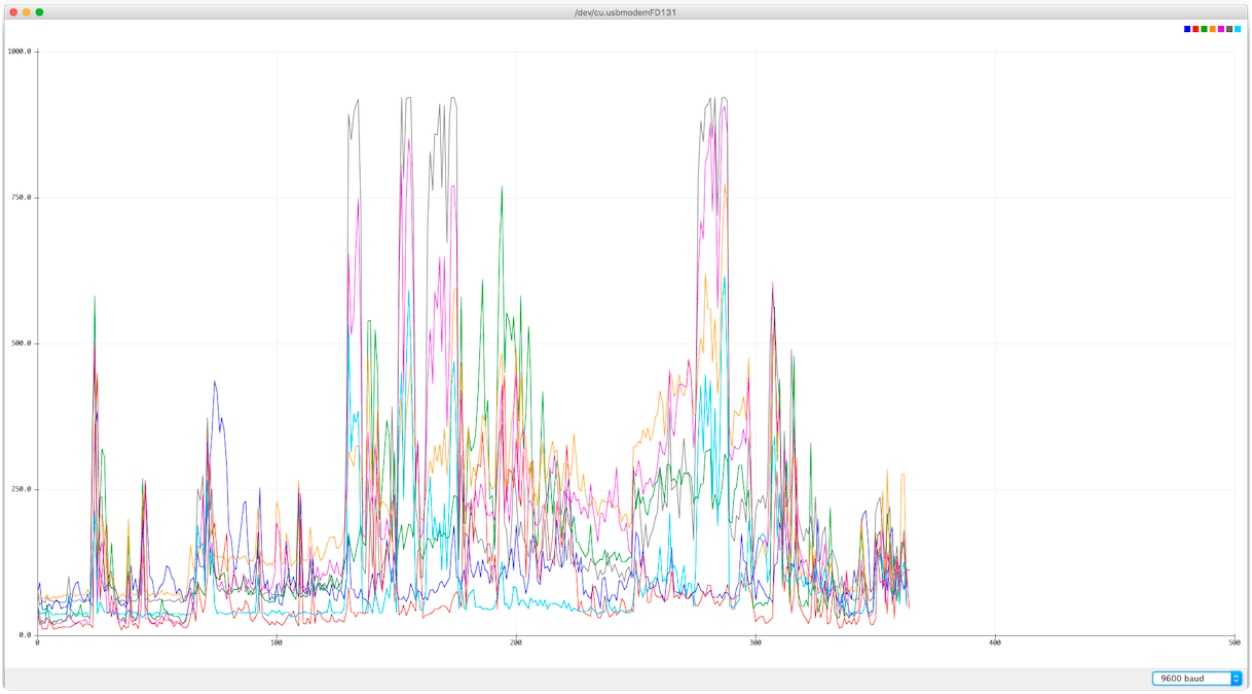
De functie `getGPS` haalt de GPS-coördinaten op en controleert deze op juistheid. Daarna wordt de batterijspanning gemeten door de functie `getBatteryVoltage` en toegevoegd aan de datastructuur die naar de cloud gaat.
Let op: als het Arduino-board wordt gevoed via VIN, is de batterijspanning nul. Alleen als hij wordt gevoed via de voedingspennen op het breadboard geeft hij de spanning weer.

Data naar Sigfox sturen

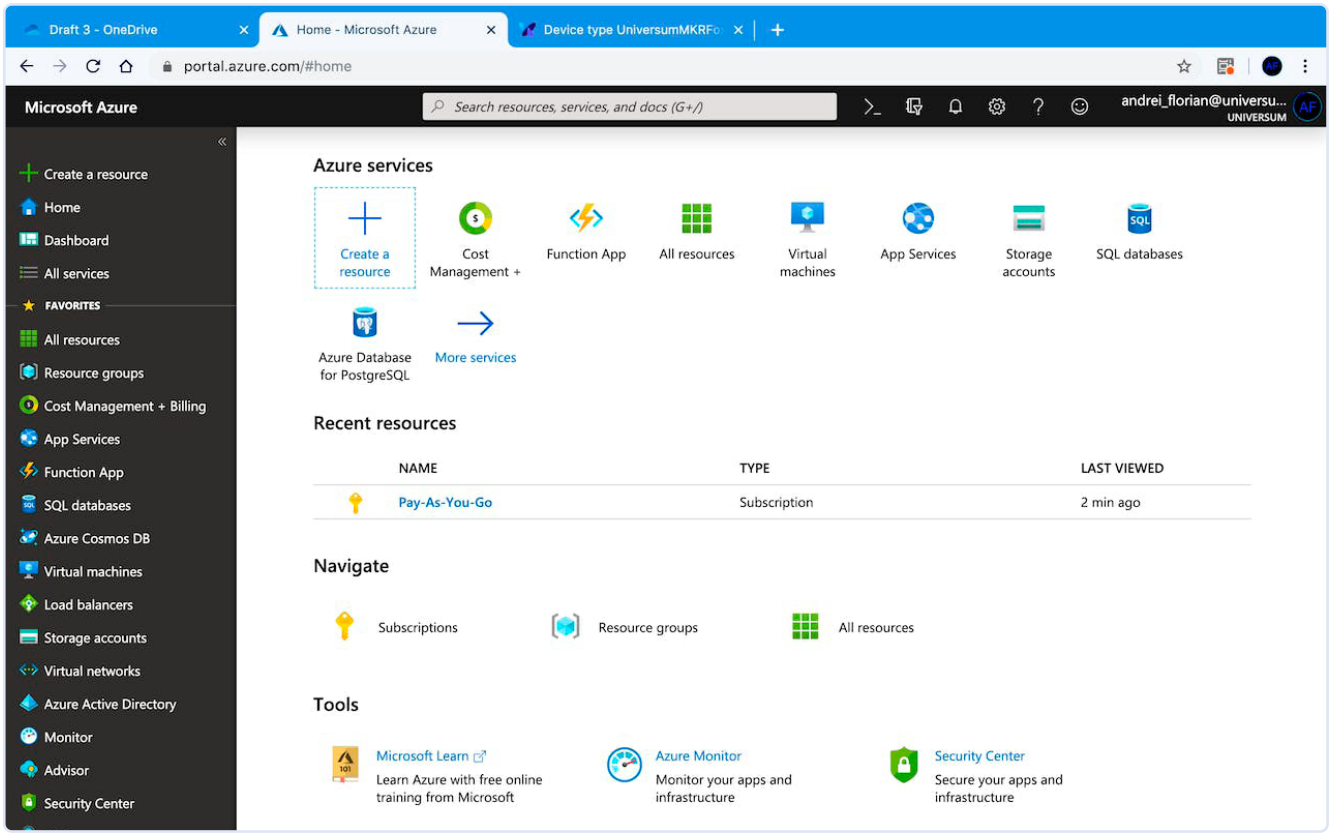
De functies `encodeData` en `sendToSigfox` coderen de positiedata en batterijspanning in byteformaat en sturen die naar het Sigfox-backend.

Configureren van het programma

Er moeten twee configuratievariabelen worden gespecificeerd:
`proDebug`: maak deze true bij het debuggen. Als deze variabele true is, moet het apparaat via een seriële verbinding worden verbonden



Figuur 5. Geluidstest: de pieken wijzen op kettingzaagactiviteit.



Figuur 6. De eerste stap van het opzetten van een Microsoft Azure-account. De negentien volgende stappen vind je bij [1].

met een computer en de seriële monitor van de Arduino-IDE moet open staan. Zet hem anders op false.
`nrSamples`: bepaalt hoeveel samples er genomen moeten worden. Elk sample bestaat uit 100 metingen.

Microsoft Azure voorbereiden

Dit project gebruikt Microsoft Azure als backend. Er zijn wel een paar dingen nodig:

- Azure-account
- Azure-abonnement
- enige kennis van de toepassing

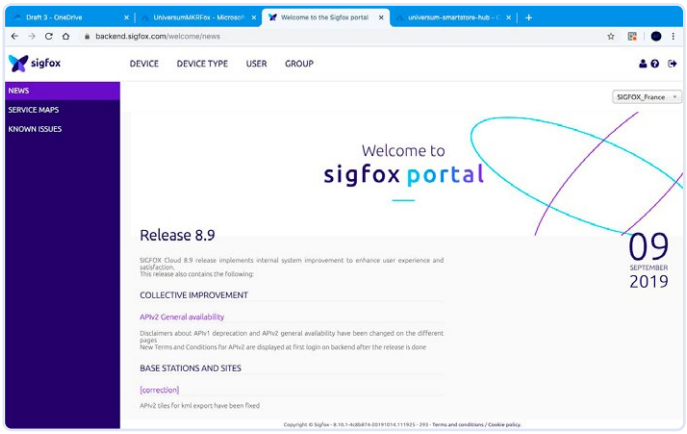
Een plaatje zegt meer dan duizend woorden (**figuur 6**). Maar twintig screenshots zouden veel te veel ruimte innemen. Ga daarom naar [1] voor details over het opzetten van een Azure-account en de IoT-hub die de data van ons apparaat moet opslaan.

Sigfox voorbereiden

We moeten ook de Sigfox-callback maken. En daar is ook weer een aantal benodigdheden:

- een Sigfox-backend-account
- registratie van het apparaat in het backend

Ook dit zou te veel plaats innemen, dus kijk naar [1] voor details over het opzetten van een Sigfox-account (**figuur 7**).
Om het Sigfox-backend te informeren over het formaat van de data die we sturen: drie floating point-getallen (breedtegraad, lengtegraad en batterijniveau) voeren we de volgende regel in in het veld



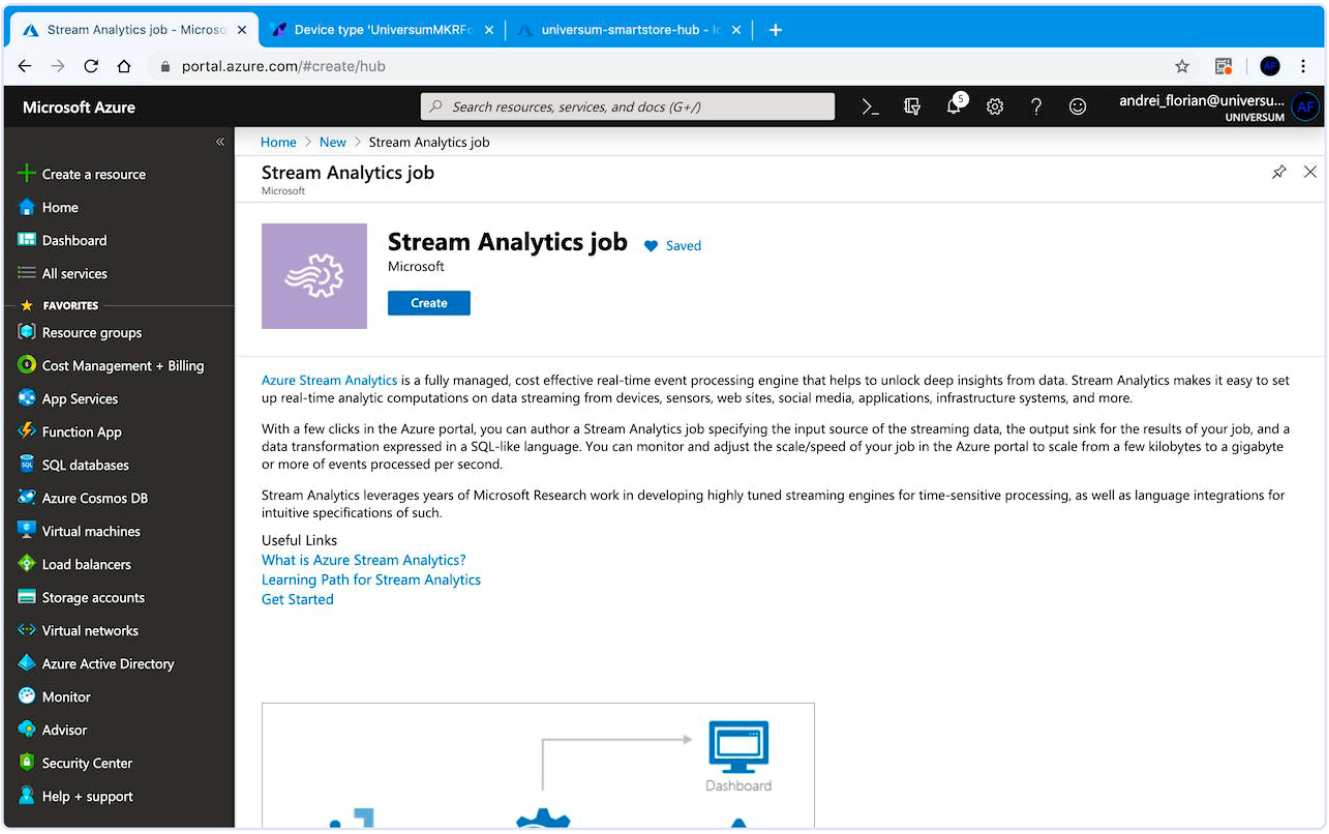
Figuur 7. Openen van het Sigfox-portaal is stap 1. De volgende negen stappen vind je bij [1].

custom-data-config:

```
geoLat::float:32:little-endian geoLng::float:32:little-endian battery::float:32:little-endian
```

Vul dan de JSON-body van het bericht met de volgende data:

```
{
    “device” : “”,
    “data” : “”,
    “latitude” : ,
```



Figuur 8. Configureer de Stream Analytics-job in Microsoft Azure.

```
    “longitude” : ,
    “battery” : ,
    “time” : ,
    “duplicate” : ,
    “snr” : ,
    “station” : “”,
    “avgSignal” : ,
    “lat” : ,
    “lng” : ,
    “rssi” : ,
    “seqNumber” :
}
```

Dit definieert de waarden die we naar Azure willen sturen.

Setup van Stream Analytics

Bij de volgende stap stellen we de Stream Analytics-job in (**figuur 8**). Deze vraagt de gegevens van de Azure IoT-hub op en stuurt die naar een Power BI-dataset. Dat lukt alleen als je eerst alle voorgaande stappen hebt doorlopen, dus zie [1] voor de details.
Bij stap 15 moet je de volgende query invullen:

```
SELECT
latitude as latitude,
longitude as longitude,
battery as battery,
System.Timestamp AS Timestamp
INTO
[OutputToPowerBI]
FROM
[InputFromIOTHub]
```

De code uploaden

Nu moeten we de code uploaden om de verbinding te testen. Zorg dat de Stream Analytics-job draait en upload de code naar het apparaat. Simuleer een kettingzaaggeluid met je telefoon en zorg dat het apparaat een event naar de cloud stuurt. De grafieken op je dashboard moeten verschuiven nadat je de data hebt verstuurd. Dat betekent dat de data is ontvangen, en dat je verder kunt gaan.
Als het event niet wordt weergegeven in de grafieken, moet je beginnen met debuggen bij het Sigfox-backend en van daar uit naar Azure toewerken.

Inrichten van Microsoft Power BI

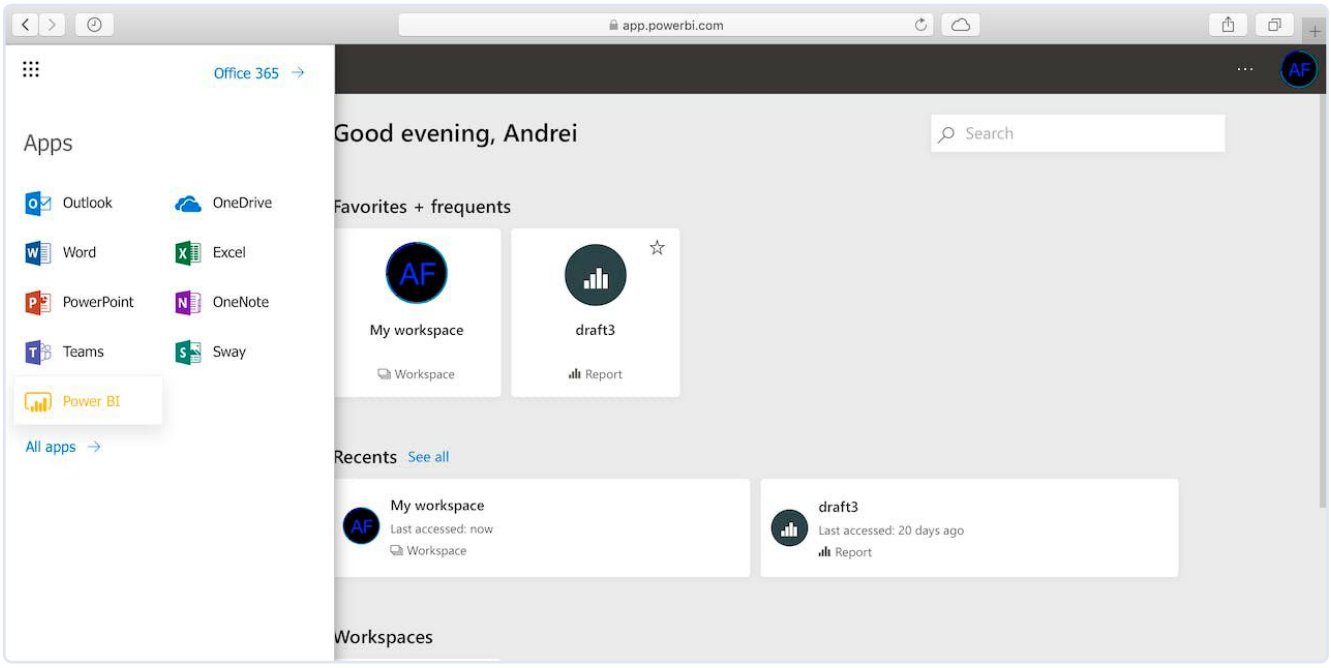
We gaan nu Microsoft Power BI (**figuur 9**) opzetten voor het visualiseren van onze data. Je hebt een zakelijk account nodig om een Power BI-abonnement toe te wijzen bij Microsoft. Hopelijk heb je dat. Anders zijn er vele alternatieven. De vereisten zijn:

- Power BI-account
- alle voorgaande stappen zijn doorlopen

Ook dit zou weer te veel plaats innemen, dus werp een blik op [1] voor details over het opzetten van een Power BI-account in vijftien stappen.

Behuizing

Als laatste stap moeten we een behuizing voor het project maken (**figuur 10**). Ik heb het apparaat aan een boom gebonden en



Figuur 9. Navigeer naar Microsoft's Power BI in de Office Online-app.

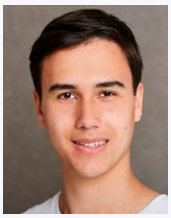
gecamoufleerd om de houthakkers te verrassen. Zorg dat de microfoon op de buitenkant zit, zodat hij het geluid kan registreren.

Een IoT-oplossing

Een array van de hier beschreven Senso-apparaten, verborgen in het bos, kan kettingzagen detecteren en de autoriteiten alarmeren als er gekapt wordt. Ik hoop dat landen met behulp van IoT-oplossingen de ontbossing een halt kunnen toeroepen. Senso is een eerste stap in het vinden van goedkope oplossingen om de klimaatverandering stap voor stap te bestrijden.

220423-03 (vertaling: Evelien Snel)

Figuur 10. De behuizing van mijn prototype.



Over de auteur

Andrei Florian is een student uit Ierland die hartstochtelijk streeft naar constructieve veranderingen en globale ontwikkeling door middel van technologie. Hij heeft aan meerdere projecten gewerkt die verband houden met de Sustainable Development Goals en is gespecialiseerd in Internet of Things en cryptografie.

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een email naar de auteur via andrei@andreiflorian.com of naar de redactie van Elektor via redactie@elektor.com.

Gerelateerde producten

Op zoek naar de belangrijkste producten die in dit artikel zijn genoemd? Arduino en Elektor hebben ze voor je klaargezet!

- **Arduino MKR FOX 1200 (SKU 19096)**
  www.elektor.nl/19096
- **OPEN-SMART GPS – Serial GPS Module for Arduino (SKU 18733)**
  www.elektor.nl/18733

WEBLINK

[1] Senso op de Arduino Project Hub: https://create.arduino.cc/projecthub/andreiflorian/senso-c00153