



**Hewlett Packard
Enterprise**

ESME
sudria
Lille - Paris - Lyon

**WE ARE
LINKYD**

Final Project

Realisation of a POC about connected electricity meters

{ TEAM }

```
def OurTeam:  
    for name in awesome_peoples:  
        print name, picture
```



Clément Tailleur



Cyril Monti



Rémi Ferreira

3 motivated **students** looking forward to become real **engineers**!

{ **CONTEXT** }

```
from IoT, SmartGrids import *  
class Context:
```



More than **50 billions** of connected objects in **2020**
Big Data's next challenge



Prices of the electricity constantly **raising** in France

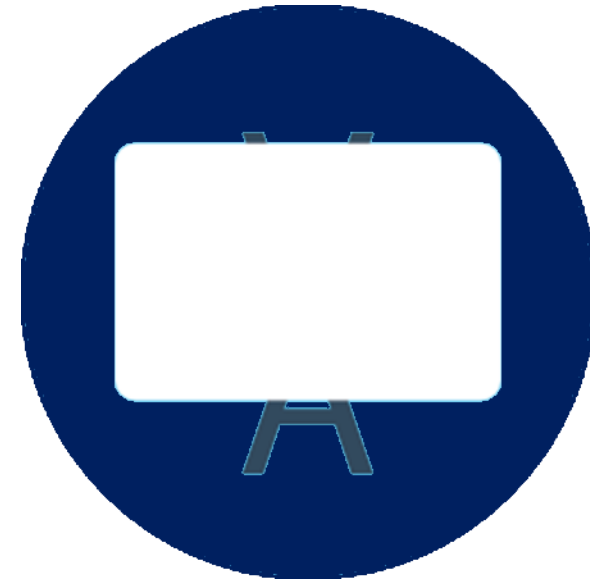
This is in this context that EDF started to launch: **Linky**

{ LINKY } | `class Linky (IoT, ElectricityMeter) :`



Electricity meter from Enedis

Planned to be deployed in **700 000 houses** and finally in **35M**

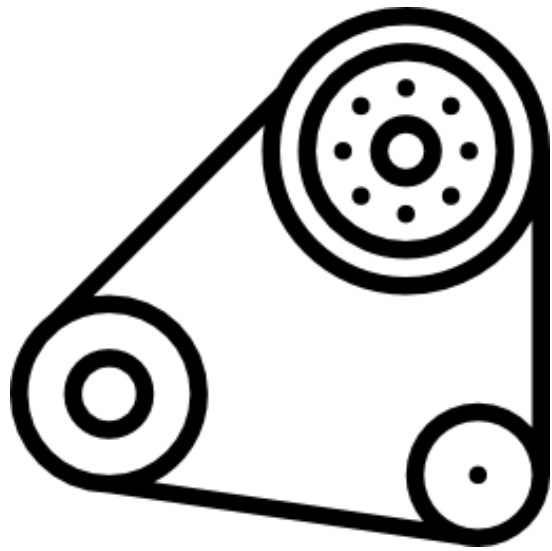


Project:

- **Simulate consumption** of 700 000 homes equipped with Linky.
 - **Store simulated data.**
- **Analyze data** and build **predictive models**

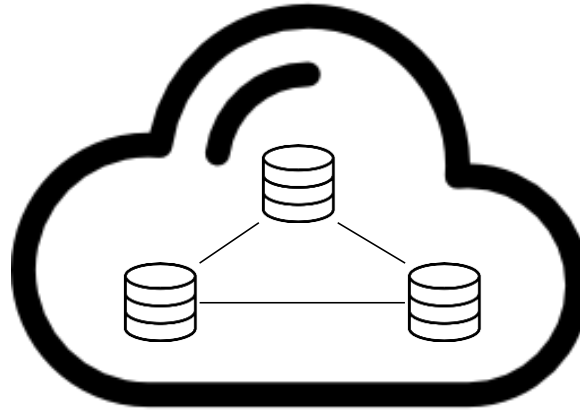
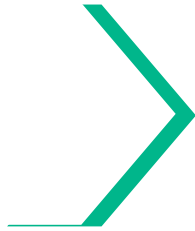
{ PROJECT }

```
project = Project(awesome_people)
```



Data Generator

Store data



Big Data Cluster

Access data



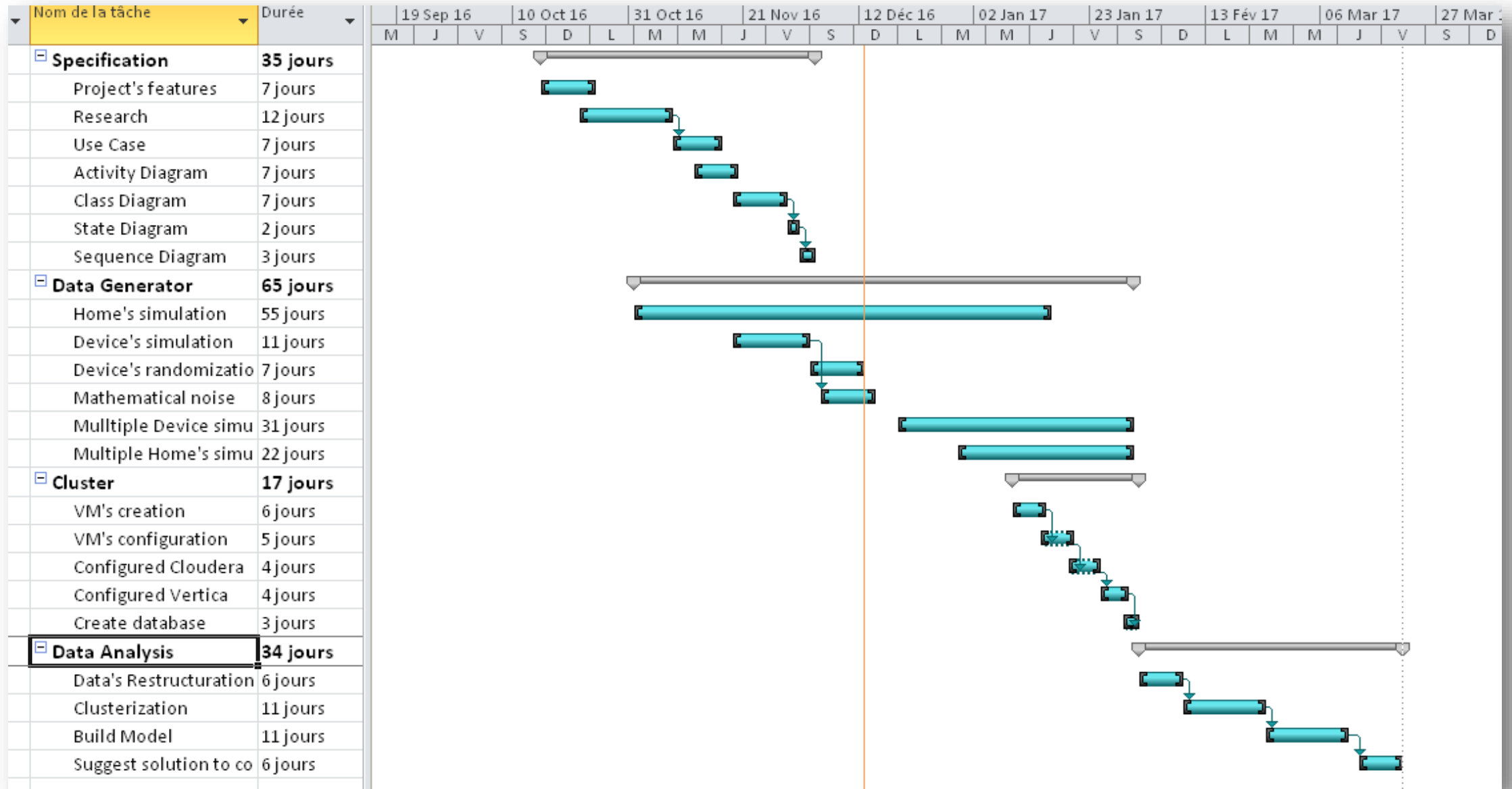
Data Analyzer



Get back initial models

{ PROJECT }

project = **Project**(awesome_people)



{ DATA GENERATOR }

```
while DataGenerator.isOn :  
    var = random(0,100)  
    totally_accurate_data.append(var)
```

Simulate:

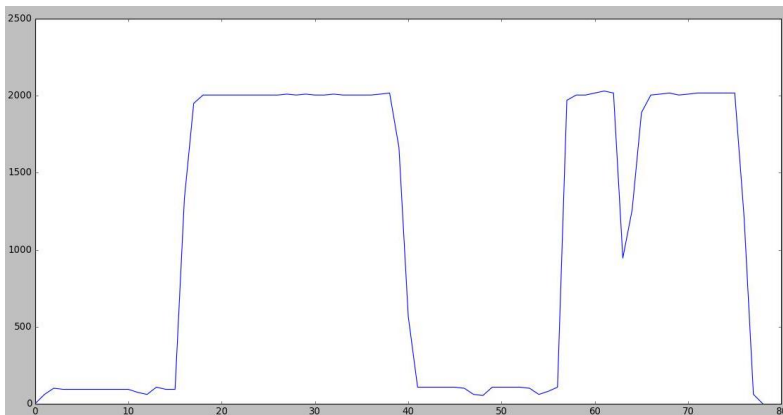


1. Consumption of devices
2. Monthly/daily usages profiles
3. A home profile
4. Consumption of a family
5. Consumption of a whole town



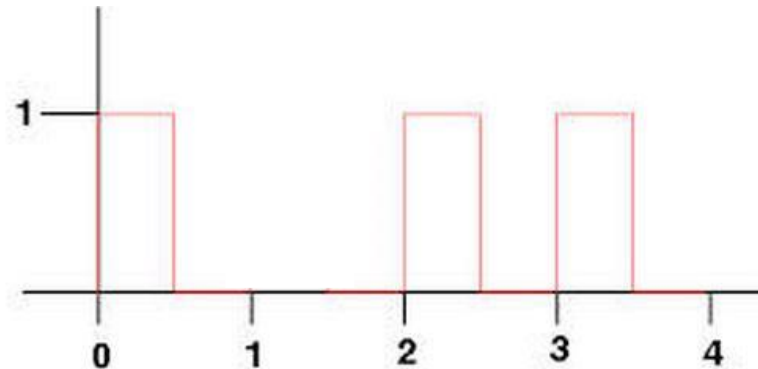
Store in database

1.



2.

X



(knowing)

3.

```
class Home:  
    nb_people = 3  
    nb_TVs = 2  
    nb_dishwasher = 1  
    .  
    .  
    .
```

{ DATA GENERATOR }

```
while DataGenerator.isOn :  
    var = random(0,100)  
    totally_accurate_data.append(var)
```

Simulate:

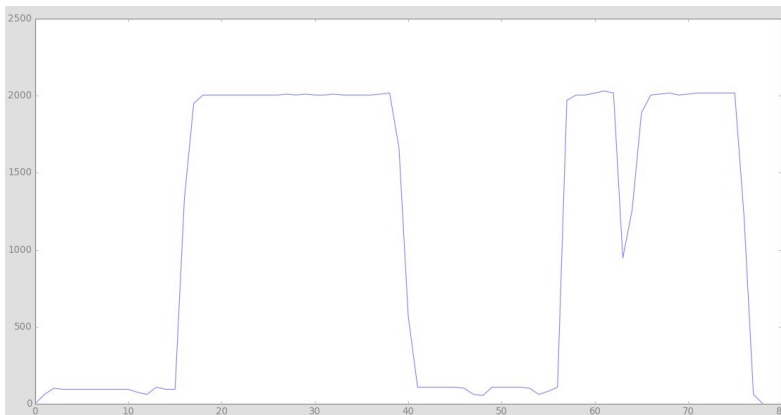


1. Consumption of devices
2. Monthly/annual usages profiles
3. A home profile
4. Consumption of a family
5. Consumption of a whole town



Store in database

1.



2.



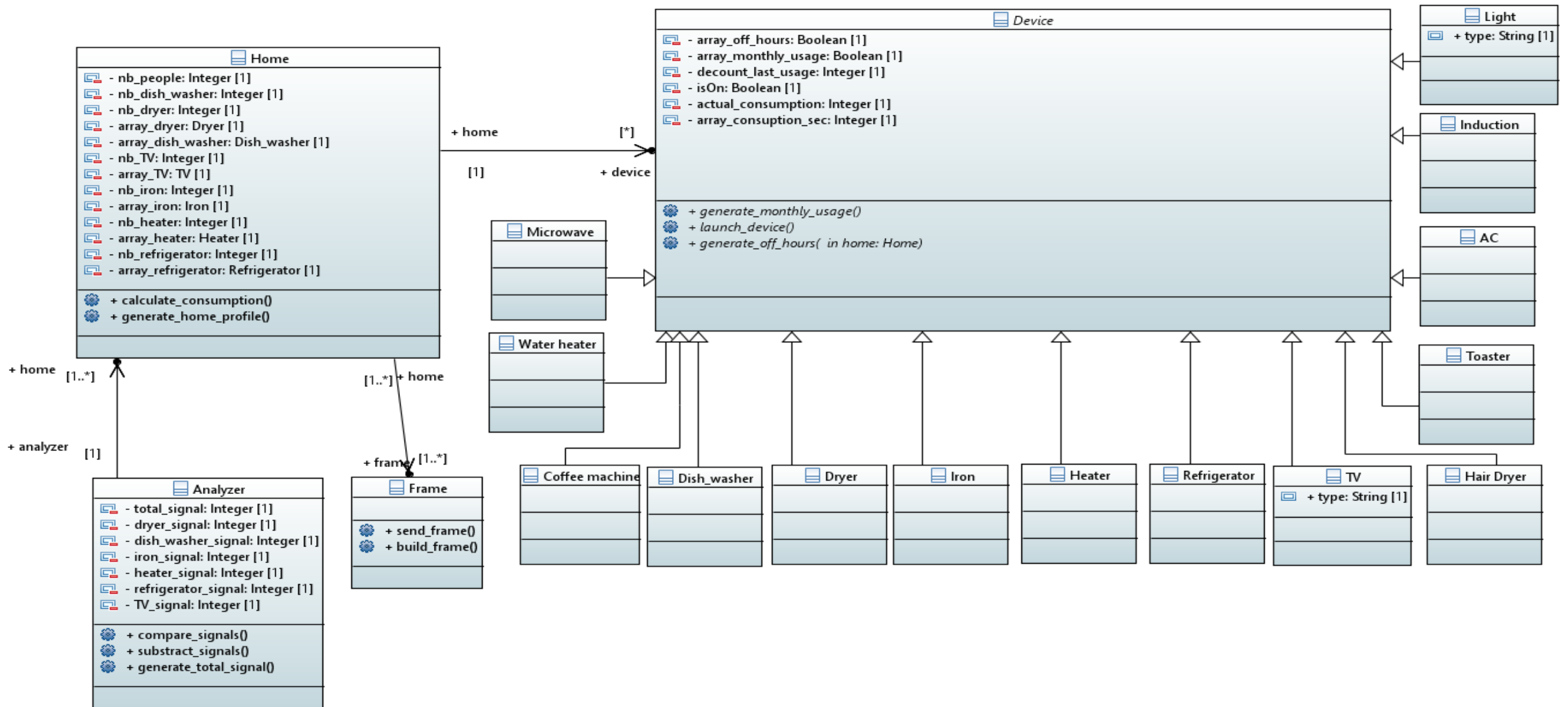
3.

```
class Home:  
    nb_people = 3  
    nb_TVs = 2  
    nb_dishwasher = 1  
    .  
    .  
    .
```

(knowing)

DATA GENERATOR

```
from CoolDiagrams
import ClassDiagram
```



{ CLUSTER }

```
cd cluster/hadoop/cloudera
```

Softwares, libraries and IDE



cloudera



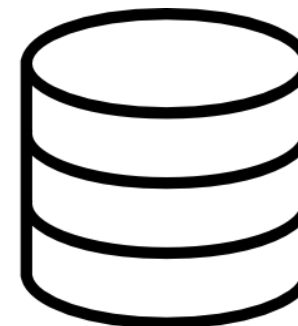
```
cd cluster/hadoop/cloudera
```



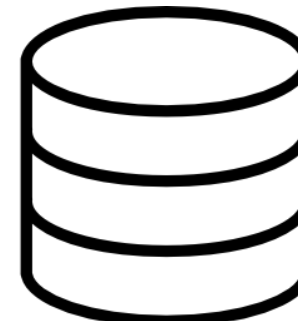
Vertica virtual machine



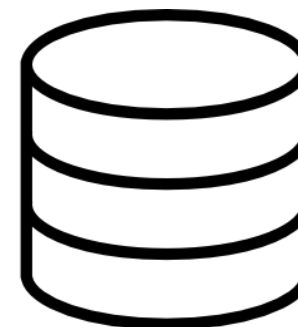
Cloudera VM - Master



Cloudera VM – Data Node 1



Cloudera VM – Data Node 2

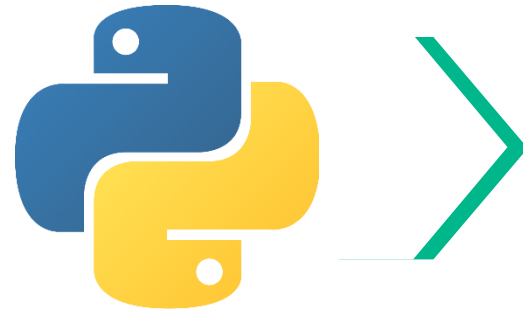


Cloudera VM – Data Node 2

{ ANALYSIS }

```
def Analysis:  
    if pattern1 == pattern2:  
        print ``Yeah, cool they are the same..``
```

Softwares, libraries and IDE



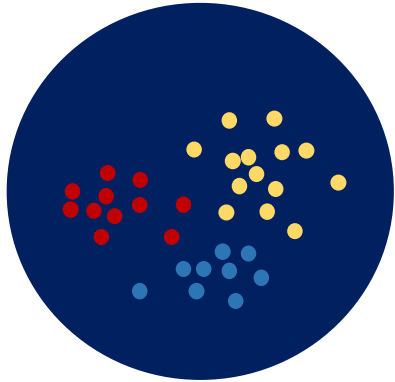
pandas
 $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



{ ANALYSIS }

```
def Analysis:  
    if pattern1 == pattern2:  
        print ``Yeah, cool they are the same..``
```

Goals



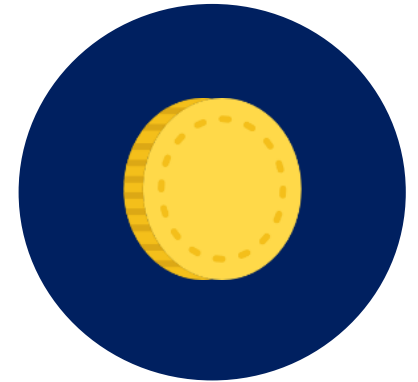
Find household type randomized
during the data generation :

clusterisation



Be able to know which device is
used at a specific point in time :

**build model based on
various households**

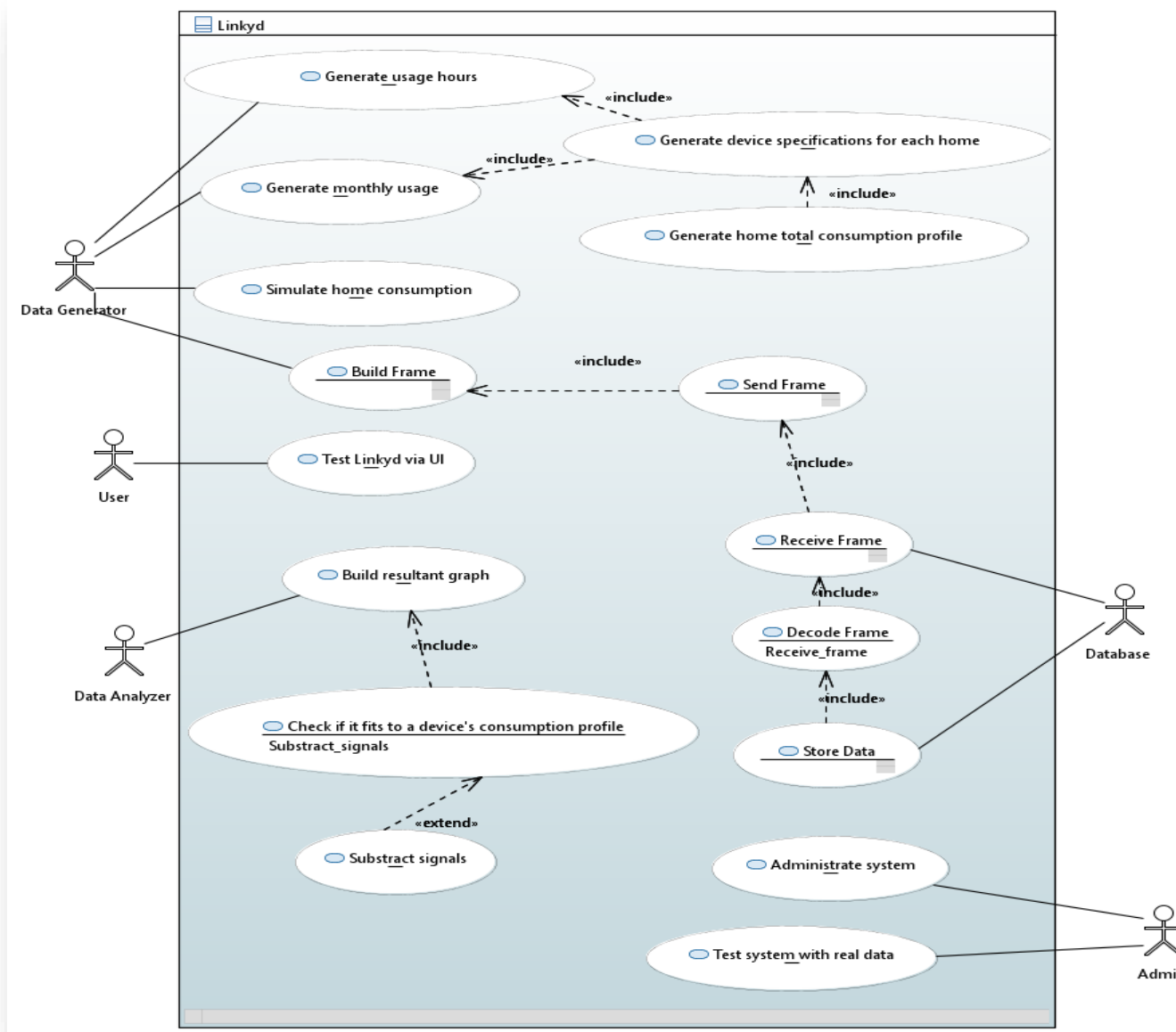


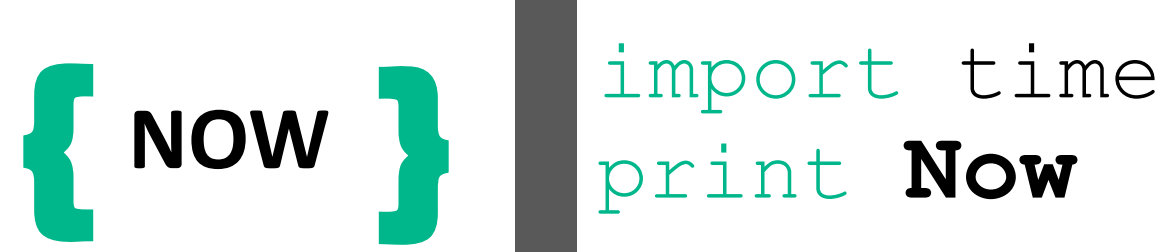
Emphasize the most electricity
consuming appliances :

**suggest solution to
consumers in order to
save money**

{ USE CASE }

```
from CoolDiagrams
import UseCaseDiagram
```





Done

- Generation of 1 device consumption signal
- Generation of a home profile
- Generation of daily/monthly usages
- Generation of a home device
- Generation of a home consumption, with one device over a year.
- Generate different signals for a single device

In process

- Finding other devices' consumption profiles
- Generating multiple devices in the same time for 1 family
- Creating VMs for the Big Data cluster

To do

- Building the cluster
- Configure data storage
- Configure cluster
- Restrcture database
- Clusterisation
- Build model based on various household
- Suggest solution to customers to optimize their consumption

{DEMO}