SIES 2022: Software Design

[Smart Integrated Energy Systems: Enhanced Virtual Power Plant VPP+ Energy Pool Integration for Local and Regional Resistance]

Gary Howorth

University of Strathclyde
Department of Electronic & Electrical Engineering

Putting together a Jigsaw Puzzle

- Learning by doing Project
- File Structure based on Prior Work (PyEMLab 2019) (Not the Only Way)
- Structured power based framework (aggregators)
- VPP –There is a both a Technical and Commercial Need for a VPP.

  Many VPP’s are Technically focused.
- Process is challenging when starting from scratch.

Design

- Generic design of VPP was set out in reference [2] and consists of a number modules including communications, accounting, forecasting, scheduling, bidding and risk management etc. (Slide 4). Initially, it was envisaged that a third party supplier would provide VPP software to communicate to and from various assets and perform many of these functions. After initial scoping of the literature and available software, few solutions providers were considered, and the most promising finally selected, with the aim of building specific functions such as forecasting and optimization in a separate prototype software environment.

Generic VPP(2009)

Octopus one way
EPEXSpot

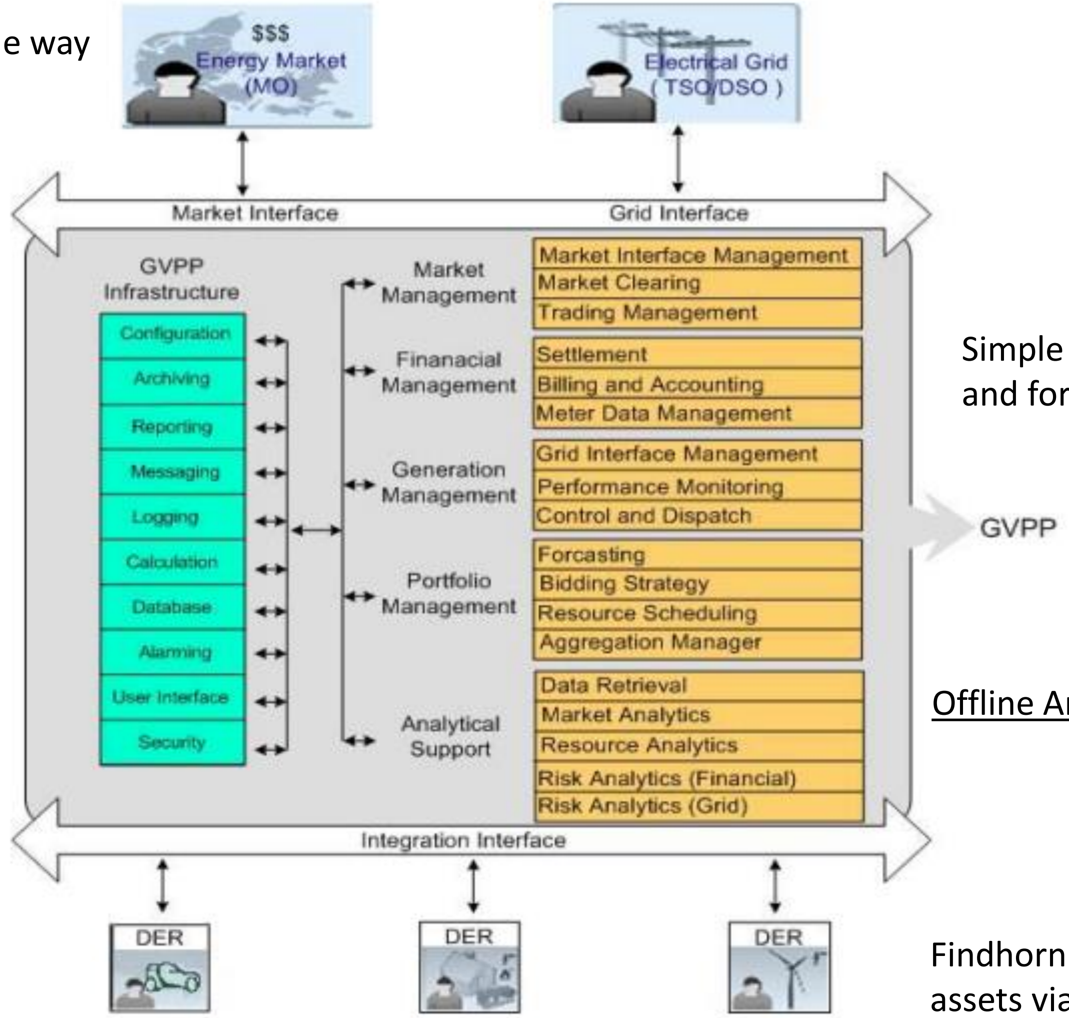


Fig. 2. Function-based Design for the GVPP

Simple Accounting Revenues
and forecast Revenues

Text alarms

Offline Analytics (market & resource)

Findhorn Myres Hill – and ETC
assets via Sequentec

S. You, C. Træholt, and B. Poulsen, "Generic Virtual Power Plants: Management of distributed energy resources under liberalized electricity market," in *8th International Conference on Advances in Power System Control, Operation and Management (APSCOM 2009)*, 2009, pp. 1-6.

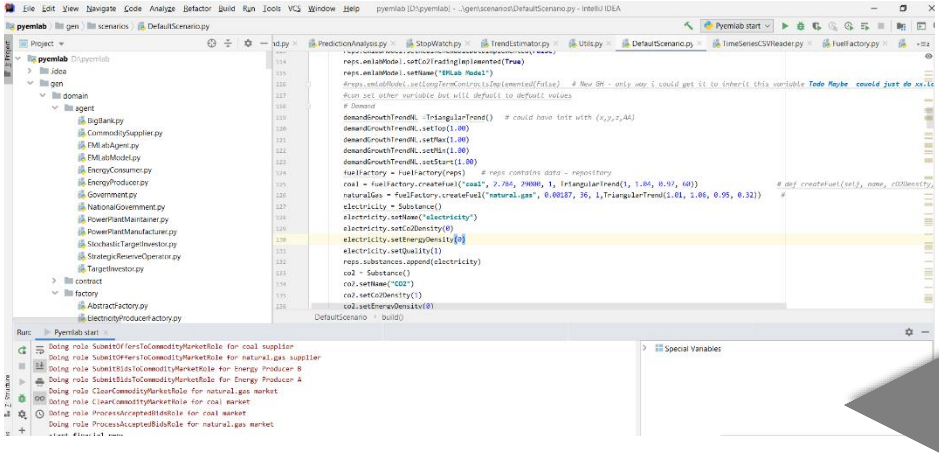
PyEMLAb-AGG

- PyEMLab-AGG [3] was developed as a python object orientated simulator to model the interactions of aggregators (VPP owners and associated actions), domestic and industrial customers in a future flexibility market. It is a structured, ontology driven, environment that uses python as a scripting language to set up scenarios and assign roles (i.e. what to do, when and how, as well as the rules to make decisions).
- It is based on a python port of the java based EMLAb program [4] which was originally designed to simulate investment and technology behaviour in an European power market. Both EMLab and PyEMLab-AGG have been extensively tested.
- The object-orientated program is organized into packages as summarized in the following slides

PyEMLab-AGG Structure

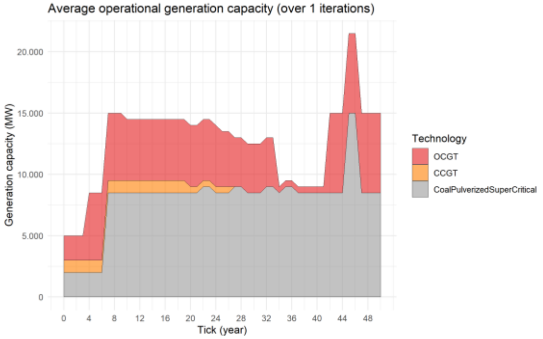
University of Strathclyde Engineering

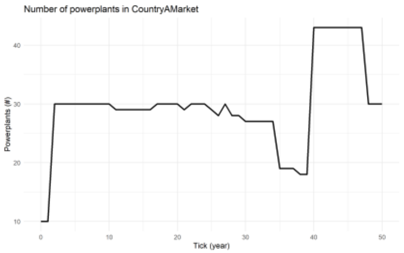
PyEmLab



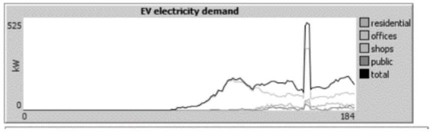
```
        reps.emlabModel.setCo2TradingImplemented(True)
        reps.emlabModel.setName("EMLab Model")
        #reps.emlabModel.setLongTermContractsImplemented(False)   # New GH - only way i could get it to inherit this variable Todo Maybe  covold just do xx.lc
        #can set other variable but will default to default values
        # Demand
        demandGrowthTrendNL =TriangularTrend()   # could have init with (x,y,z,AA)
        demandGrowthTrendNL.setTop(1.00)
        demandGrowthTrendNL.setMax(1.00)
        demandGrowthTrendNL.setMin(1.00)
        demandGrowthTrendNL.setStart(1.00)
        fuelFactory = FuelFactory(reps)    # reps contains data - repository
        coal = fuelFactory.createFuel("coal", 2.784, 29000, 1, TriangularTrend(1, 1.04, 0.97, 60))          # def createFuel(self, name, cO2Density,
        naturalGas = fuelFactory.createFuel("natural.gas", 0.00187, 36, 1,TriangularTrend(1.01, 1.06, 0.95, 0.32))    #
        electricity = Substance()
        electricity.setName("electricity")
        electricity.setCo2Density(0)
        electricity.setEnergyDensity(0)
        electricity.setQuality(1)
        reps.substances.append(electricity)
        co2 = Substance()
        co2.setName("CO2")
        co2.setCo2Density(1)
        co2.setEnergyDensity(0)
```

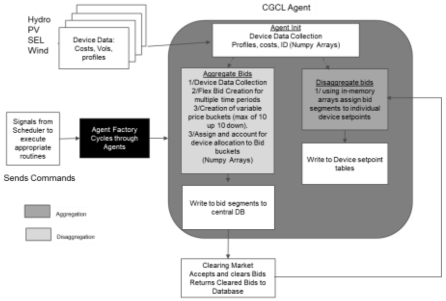
```
Doing role SubmitOffersToCommodityMarketRole for coal supplier
Doing role SubmitOffersToCommodityMarketRole for natural.gas supplier
Doing role SubmitBidsToCommodityMarketRole for Energy Producer B
Doing role SubmitBidsToCommodityMarketRole for Energy Producer A
Doing role ClearCommodityMarketRole for natural.gas market
Doing role ClearCommodityMarketRole for coal market
Doing role ProcessAcceptedBidsRole for coal market
Doing role ProcessAcceptedBidsRole for natural.gas market
```


Average operational generation capacity (over 1 iterations)


Number of powerplants in CountryAMarket

Extensions


EV electricity demand


CGCL Agent

Hydro
PV
SEL
Wind

Device Data: Costs, Vols, profiles

Agent Init
Device Data Collection
Profiles, costs, ID (Numpy Arrays)

Aggregate Bids
1/Device Data Collection
2/Flex Bid Creation for multiple time periods
3/Creation of variable price buckets (max of 10 up 10 down).
3/Assign and account for device allocation to Bid buckets
(Numpy Arrays)

Disaggregate bids
1/ using in-memory arrays assign bid segments to individual device setpoints

Signals from Scheduler to execute appropriate routines

Agent Factory Cycles through Agents

Sends Commands

Write to bid segments to central DB

Write to Device setpoint tables

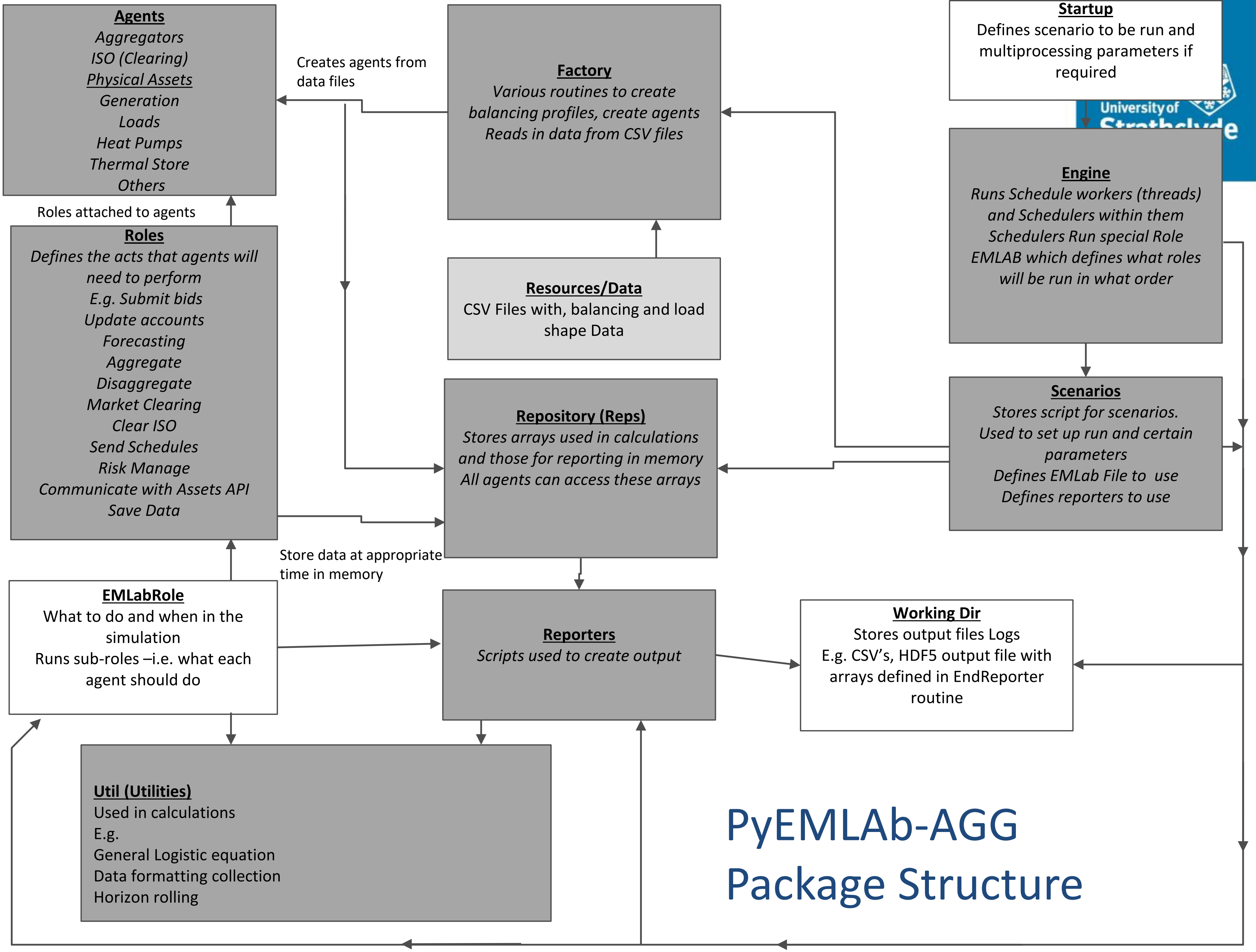
Aggregation

Disaggregation

Clearing Market
Accepts and clears Bids
Returns Cleared Bids to Database

Kiel
Lübeck
Schwerin
Hamburg
Bremen
Groningen
Niedersachsen
Nederland
Amsterdam
Utrecht
Apeldoorn
Arnhem
Hannover
Wolfsburg
Braunschweig
Magdeburg
Hildesheim
Bielefeld
Münster
Paderborn
Sachsen-Anhalt
Leaflet

PyEMLAb-AGG Package Structure



Startup
Defines scenario to be run and multiprocessing parameters if required

Agents
Aggregators
ISO (Clearing)
Physical Assets
Generation
Loads
Heat Pumps
Thermal Store
Others

Creates agents from data files

Factory
Various routines to create balancing profiles, create agents
Reads in data from CSV files

Roles attached to agents

Roles
Defines the acts that agents will need to perform
E.g. Submit bids
Update accounts
Forecasting
Aggregate
Disaggregate
Market Clearing
Clear ISO
Send Schedules
Risk Manage
Communicate with Assets API
Save Data

Resources/Data
CSV Files with, balancing and load shape Data

Engine
Runs Schedule workers (threads) and Schedulers within them
Schedulers Run special Role EMLAB which defines what roles will be run in what order

Repository (Reps)
Stores arrays used in calculations and those for reporting in memory
All agents can access these arrays

Scenarios
Stores script for scenarios.
Used to set up run and certain parameters
Defines EMLab File to use
Defines reporters to use

Store data at appropriate time in memory

EMLabRole
What to do and when in the simulation
Runs sub-roles –i.e. what each agent should do

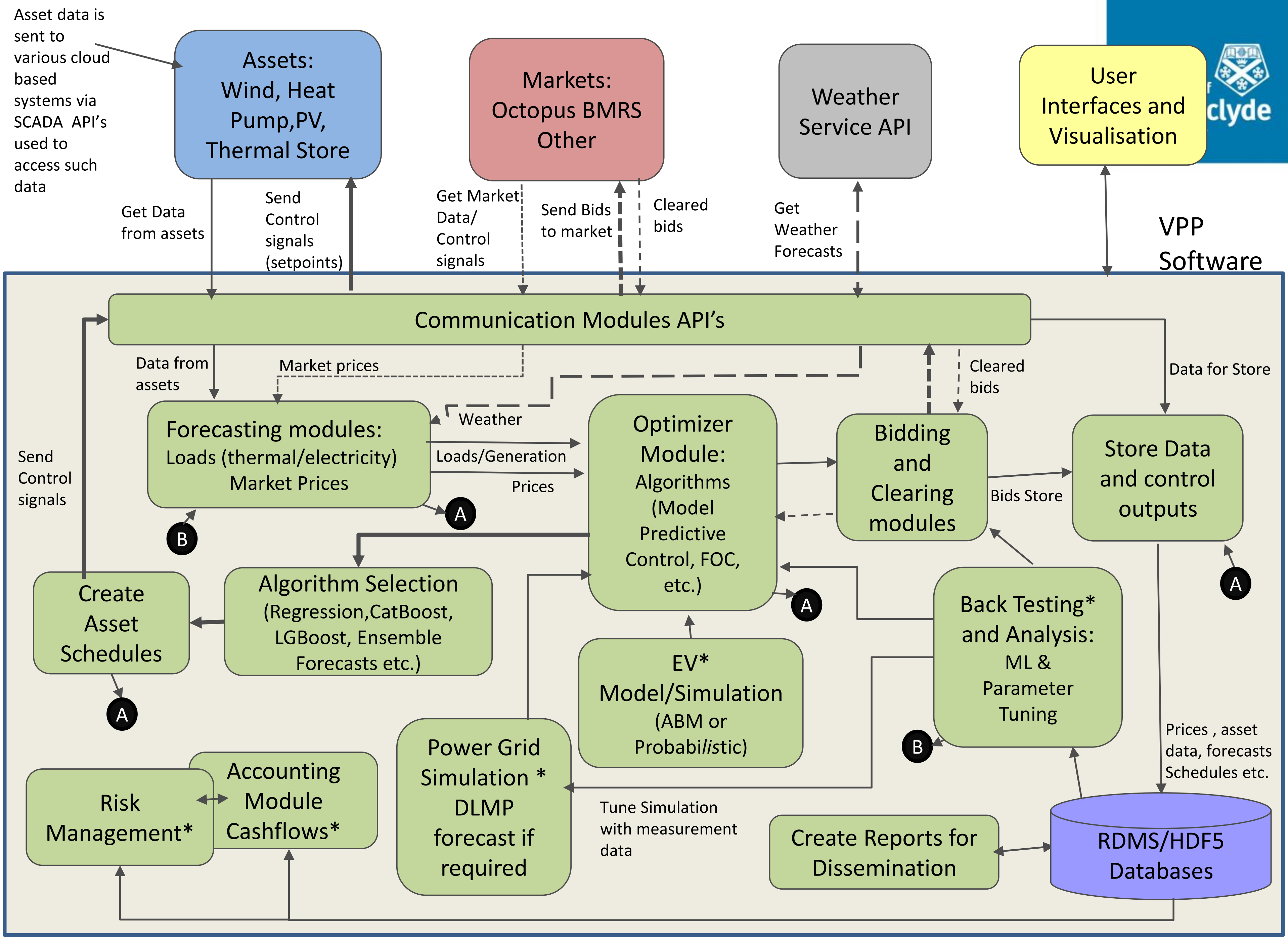
Reporters
Scripts used to create output

Working Dir
Stores output files Logs
E.g. CSV's, HDF5 output file with arrays defined in EndReporter routine

Util (Utilities)
Used in calculations
E.g.
General Logistic equation
Data formatting collection
Horizon rolling

Additional Development

- Using the PyEMLab-AGG structure discussed above, the framework has been rebuilt to communicate in real time every half hour with assets at a number of locations including the ETC site (Slide 9).
- This was performed in a step wise fashion adding and testing functionality using use cases as a guide. The first use case was based on one asset with a simple control rule. Later use cases involve additional assets such as heat pumps/thermal stores and the interaction with more complicated market structures. The current architecture for the software framework is shown in Slide 9 Those modules marked with an asterix* are for future development.
- A sequence diagram (Slide 10) shows the interactions between various components for a simple use case.



Asset data is sent to various cloud based systems via SCADA API's used to access such data

Assets: Wind, Heat Pump,PV, Thermal Store

Markets: Octopus BMRS Other

Weather Service API

User Interfaces and Visualisation

Get Data from assets

Send Control signals (setpoints)

Get Market Data/ Control signals

Send Bids to market

Cleared bids

Get Weather Forecasts

VPP Software

Communication Modules API's

Data from assets

Market prices

Cleared bids

Data for Store

Weather

Forecasting modules: Loads (thermal/electricity) Market Prices

Loads/Generation

Prices

Optimizer Module: Algorithms (Model Predictive Control, FOC, etc.)

Bidding and Clearing modules

Bids Store

Store Data and control outputs

Send Control signals

A

B

Create Asset Schedules

Algorithm Selection (Regression,CatBoost, LGBoost, Ensemble Forecasts etc.)

A

EV* Model/Simulation (ABM or Probabi*listic*)

Back Testing* and Analysis: ML & Parameter Tuning

A

B

A

Prices , asset data, forecasts Schedules etc.

Risk Management*

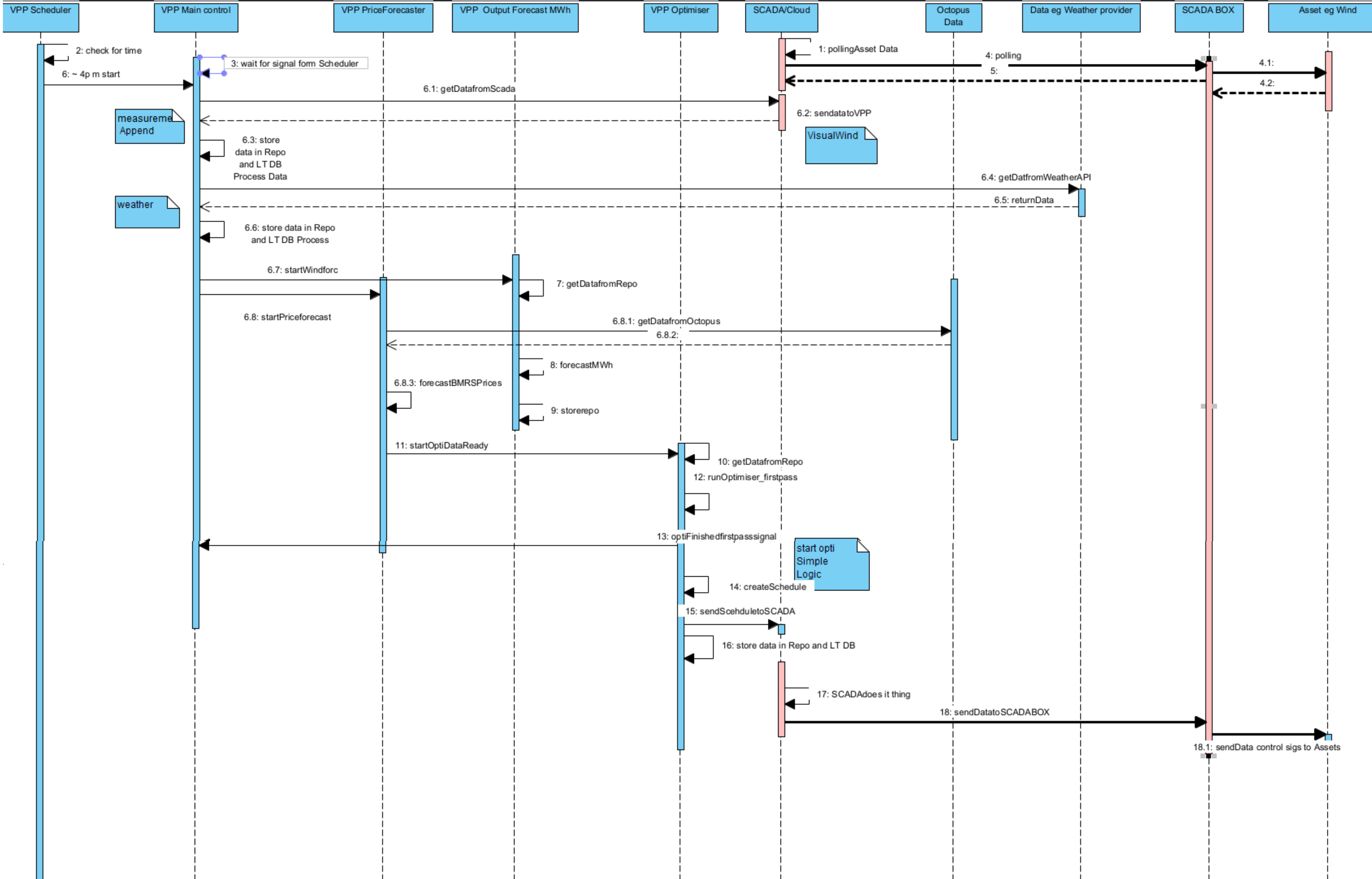
Accounting Module Cashflows*

Power Grid Simulation * DLMP forecast if required

Tune Simulation with measurement data

Create Reports for Dissemination

RDMS/HDF5 Databases



VPP Scheduler

VPP Main control

VPP PriceForecaster

VPP Output Forecast MWh

VPP Optimiser

SCADA/Cloud

Octopus Data

Data eg Weather provider

SCADA BOX

Asset eg Wind

1: pollingAsset Data

2: check for time

3: wait for signal form Scheduler

4: polling

4.1:

4.2:

5:

6: ~ 4p m start

6.1: getDatafromScada

6.2: sendatatoVPP

measureme Append

VisualWind

6.3: store data in Repo and LT DB Process Data

6.4: getDatfromWeatherAPI

6.5: returnData

weather

6.6: store data in Repo and LT DB Process

6.7: startWindforc

7: getDatafromRepo

6.8: startPriceforecast

6.8.1: getDatafromOctopus

6.8.2:

8: forecastMWh

6.8.3: forecastBMRSPrices

9: storerepo

11: startOptiDataReady

10: getDatafromRepo

12: runOptimiser_firstpass

13: optiFinishedfirstpasssignal

start opti Simple Logic

14: createSchedule

15: sendScehduletoSCADA

16: store data in Repo and LT DB

17: SCADAdoes it thing

18: sendDatatoSCADABOX

18.1: sendData control sigs to Assets

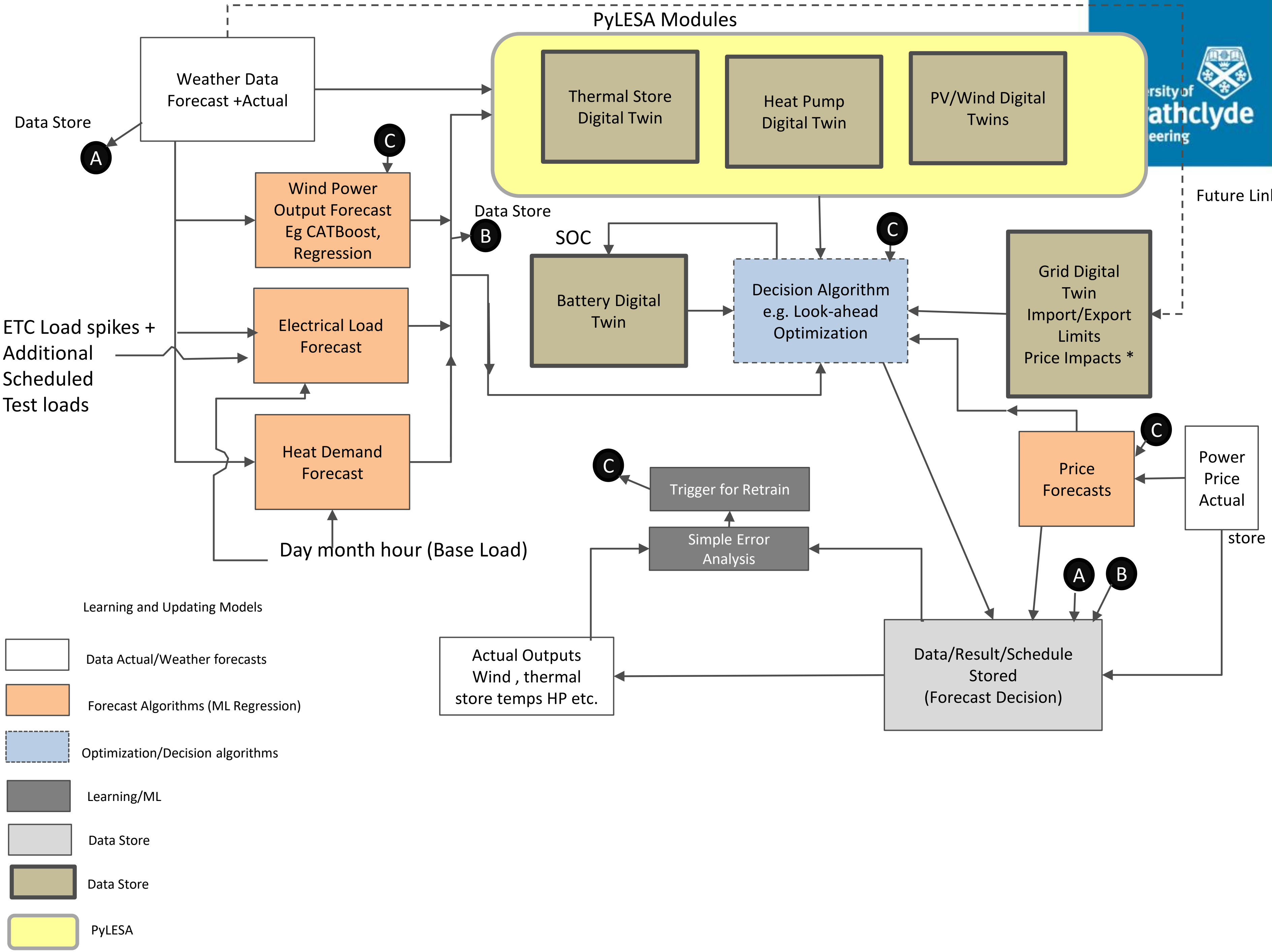
Simple Use Case: Sequence Diagram

Communications

- A key role of the model is to communicate with the assets in the field. This is achieved using API’s, some of which have had to be developed for this project.
- Data is collected and stored for later use, but those that are needed for immediate use are also stored into in-memory storage in the repository object described earlier.
- PDF style reports can be created and can be emailed to the appropriate parties. The software uses a rolling time horizon to forecast prices and demand, and is used in the decision module.

VPP Forecasting

- An important component of this VPP platform is associated with forecasting. Current VPP design uses deterministic algorithms but stochastic algorithms and approaches are being investigated.
- See slide 13 for schematic of forecasting modules.
- A number of forecasting modules have been constructed, some of which use existing machine learning libraries such as CatBoost [9], others that use standard regression techniques. Ensemble learning [10] could be included in a future version.
- In addition, PyLESA [11] an open source modelling tool for the design of local, integrated and smart energy systems and includes calculations and modules for solar/wind assets, heat pumps and thermal stores (digital twins) can also be included. It also uses the Gekko [12] optimisation model to perform forward looking model predictive and fixed order control of the assets. Various components of PyLESA have been integrated into the modified PyEMLab framework and are to be used to optimise heat pump and thermal store assets.



PyLESA Modules

Weather Data Forecast +Actual

Data Store

A

Thermal Store Digital Twin

Heat Pump Digital Twin

PV/Wind Digital Twins

Future Lin

C

Wind Power Output Forecast Eg CATBoost, Regression

Data Store

B

SOC

Battery Digital Twin

C

Decision Algorithm e.g. Look-ahead Optimization

Grid Digital Twin Import/Export Limits Price Impacts *

ETC Load spikes + Additional Scheduled Test loads

Electrical Load Forecast

Heat Demand Forecast

Day month hour (Base Load)

C

Trigger for Retrain

Simple Error Analysis

C

Price Forecasts

Power Price Actual

store

A

B

Actual Outputs Wind , thermal store temps HP etc.

Data/Result/Schedule Stored (Forecast Decision)

Learning and Updating Models

□	Data Actual/Weather forecasts
🟧	Forecast Algorithms (ML Regression)
▭	Optimization/Decision algorithms
■	Learning/ML
▨	Data Store
▣	Data Store
🟨	PyLESA

Optimization

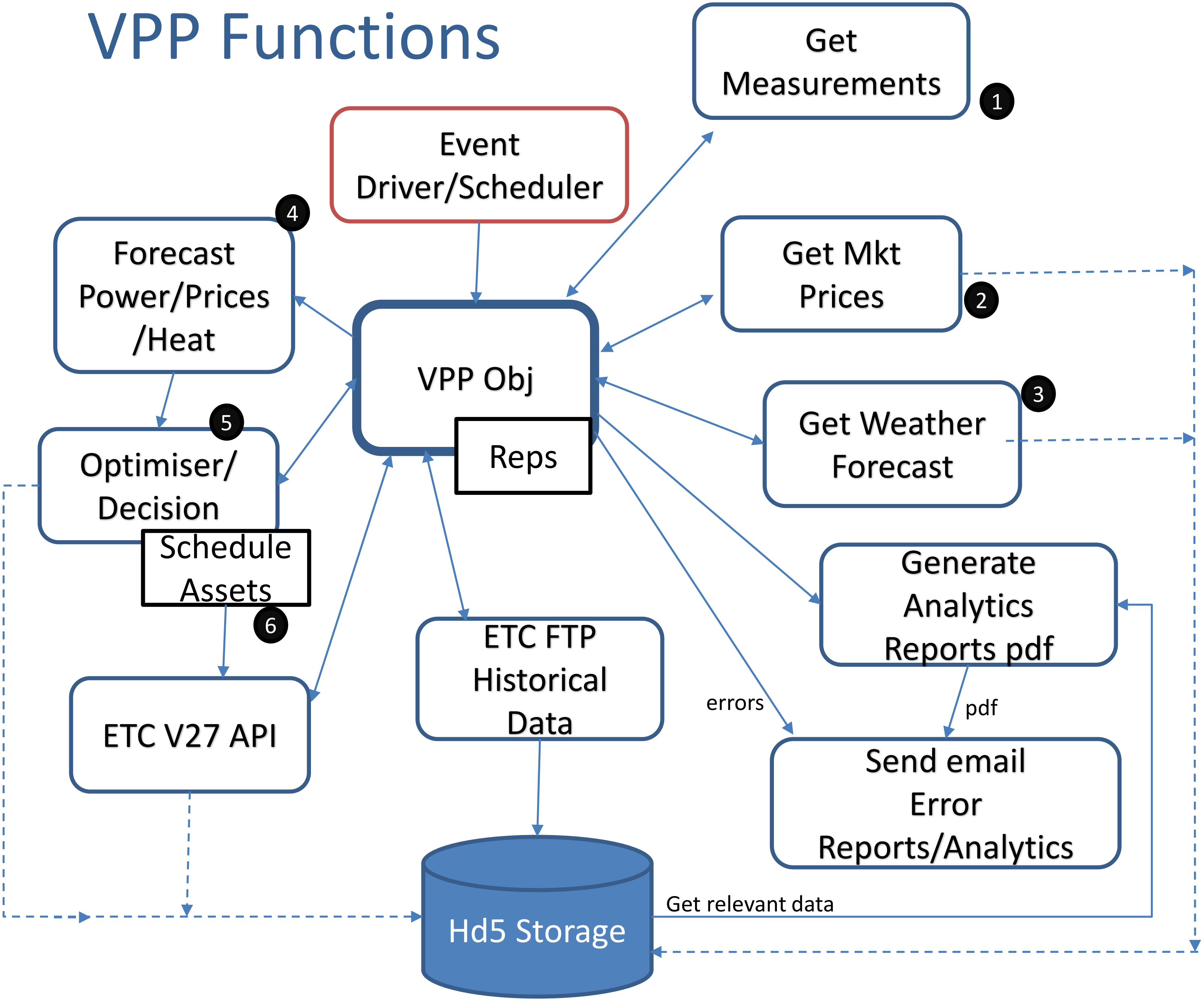
- A decision optimization model, based on the work in reference [6, 7], is included, but has been adapted to include battery degradation costs import/export limits and carbon prices. It uses the Pyomo [8] optimization model. This has been further adapted to include some elements of the current UK flexibility auction market. The optimizer, or decision model, looks to maximize revenues to the project and formulates schedules, which are then sent via the communication module to the various assets. The current model uses Octopus market prices so assumes the
- The current VPP is a price taker. This means that that bidding module is not currently used but would be as other markets are accessed. The decision module also includes other algorithms such as load following or other simple heuristics such as buy-low-sell-high and so on. Additional algorithms can be included. Note that analysis of algorithms and data is currently carried out offline.

VPP Operation

- Acquire Data
- Forecast Future Demand, Power Output , Export Import Limits etc.
- Decision Process (Simple Heuristics , FOC, Optimization other)
- Schedule Assets
- Learn

- Example Operation shown below using One Asset

VPP Functions



Get Measurements 1

Event Driver/Scheduler

Forecast Power/Prices /Heat 4

VPP Obj

Reps

Get Mkt Prices 2

Get Weather Forecast 3

Optimiser/ Decision 5

Schedule Assets 6

ETC V27 API

ETC FTP Historical Data

Generate Analytics Reports pdf

pdf

errors

Send email Error Reports/Analytics

Hd5 Storage

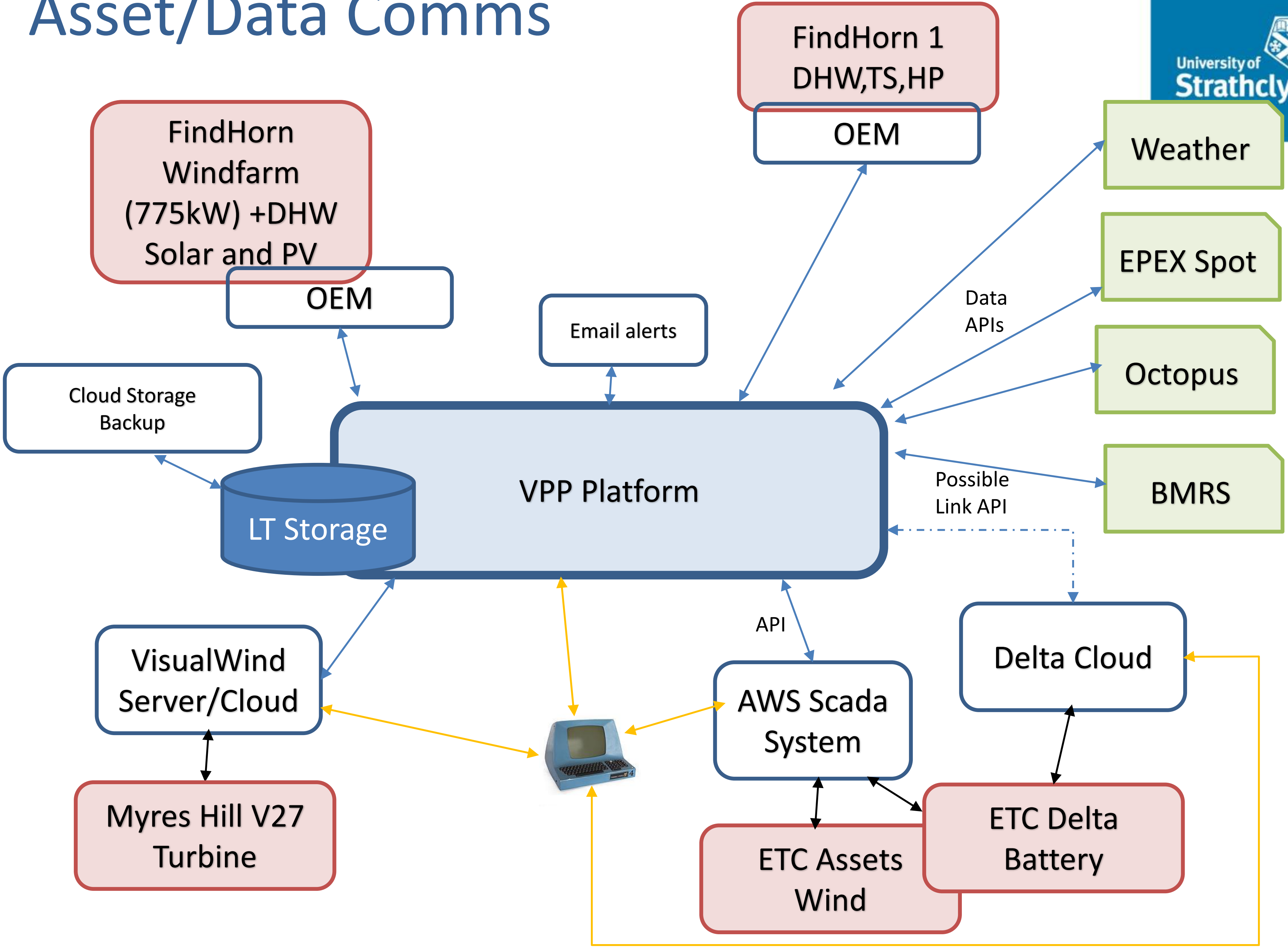
Get relevant data

Store data for long term use and Analysis

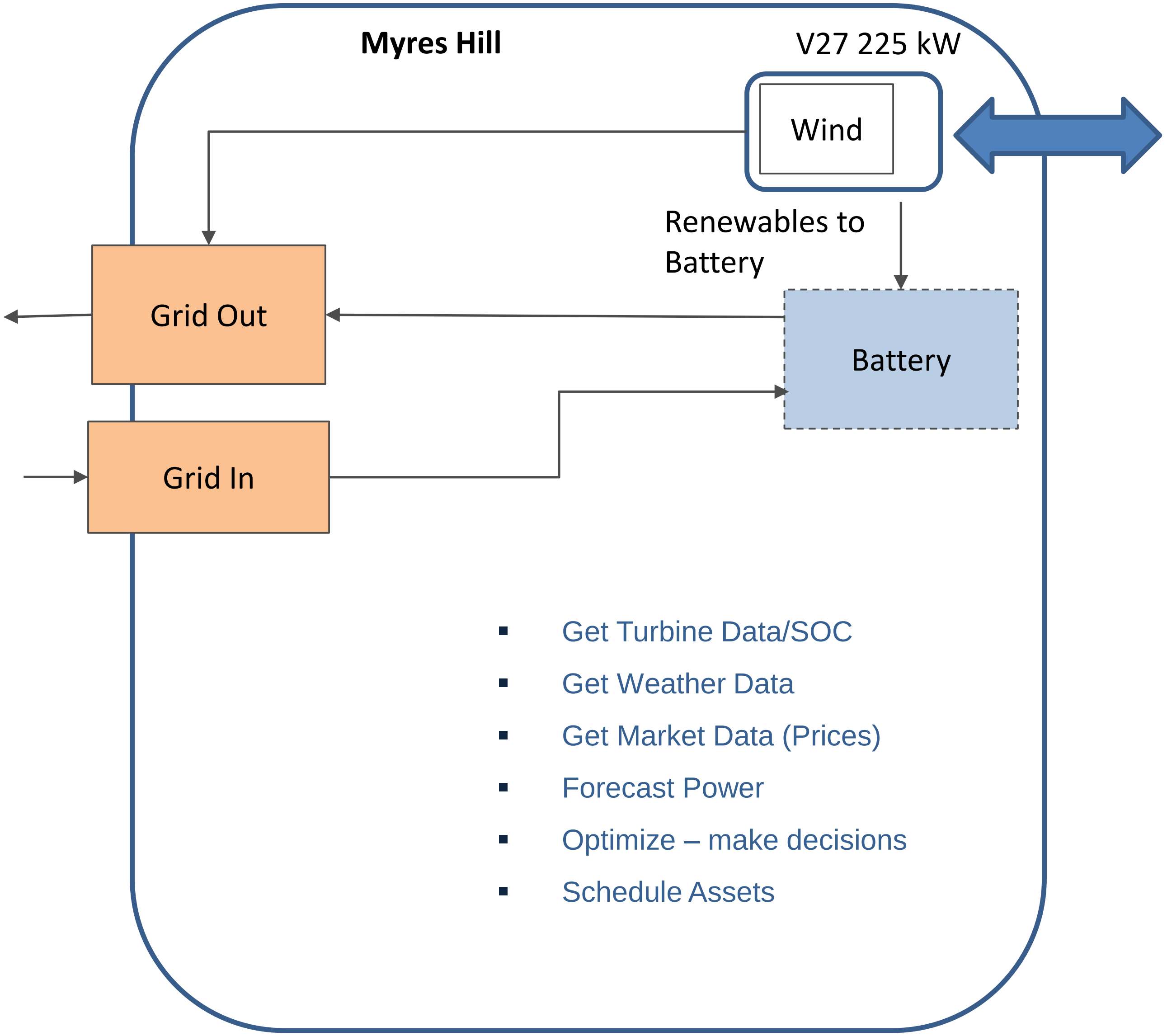
VPP ACT:

- Act at certain times
- Do certain things
- Use obj attached
- Obj stores status flags and settings

Asset/Data Comms



Simple Use Case Example

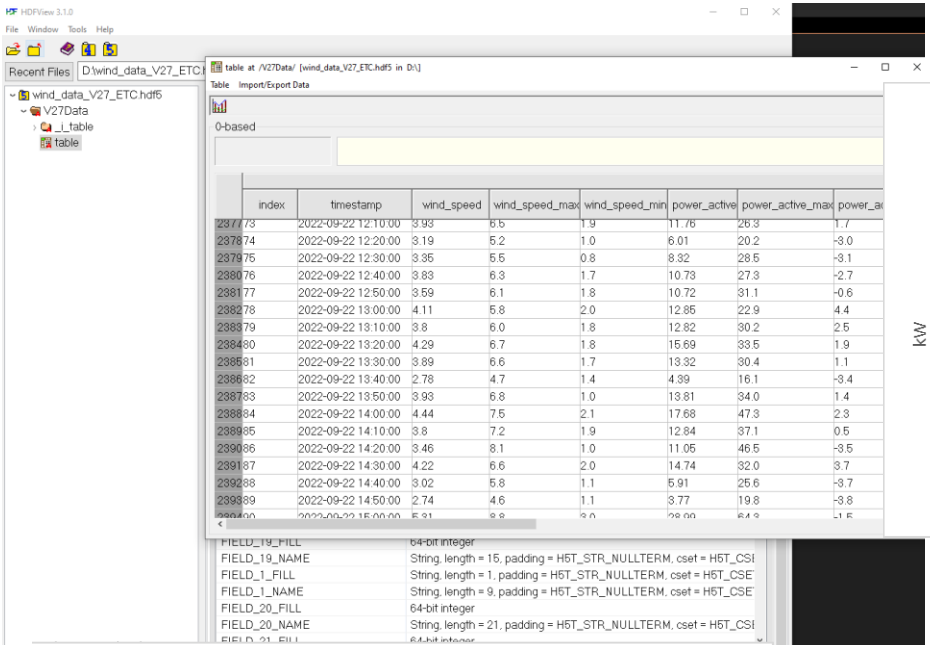


API to Communicate to from Wind Turbine

- Get Turbine Data/SOC
- Get Weather Data
- Get Market Data (Prices)
- Forecast Power
- Optimize – make decisions
- Schedule Assets

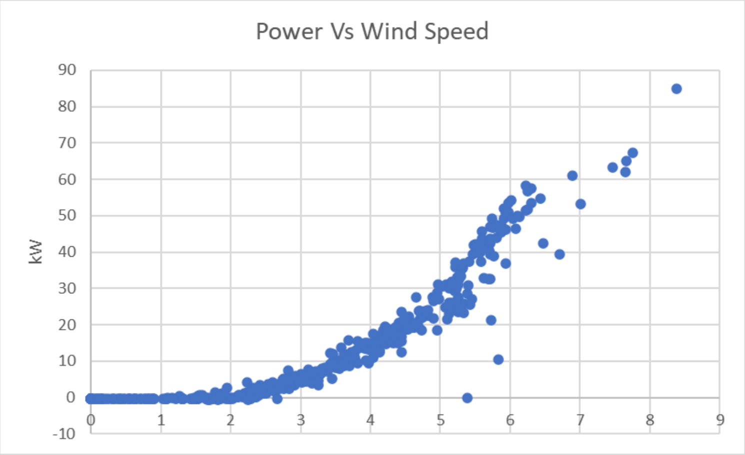
Get Measurement Data

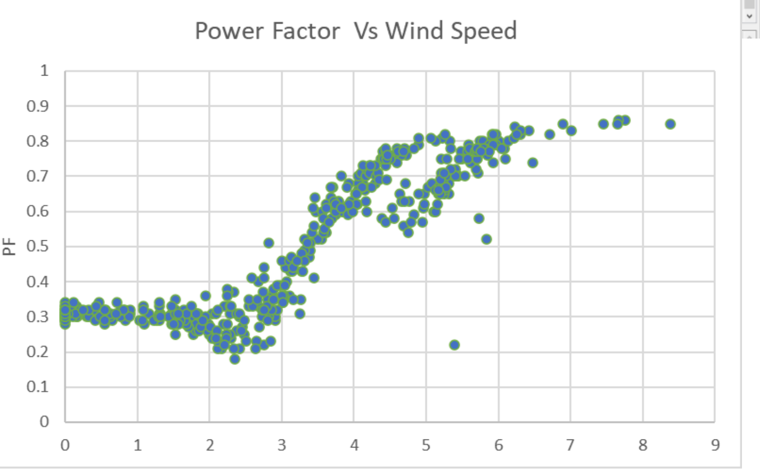
1

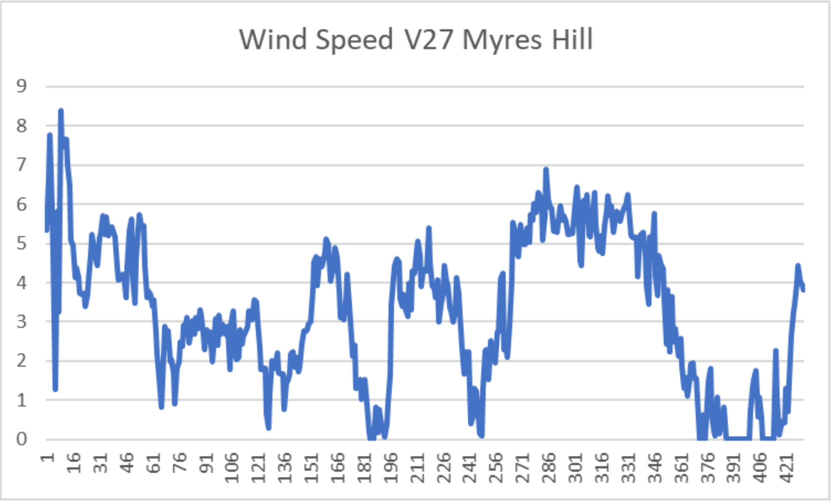


	index	timestamp	wind_speed	wind_speed_max	wind_speed_min	power_active	power_active_max	power_ac
2377	73	2022-09-22 12:10:00	3.93	6.5	1.9	11.76	26.3	1.7
2378	74	2022-09-22 12:20:00	3.19	5.2	1.0	6.01	20.2	-3.0
2379	75	2022-09-22 12:30:00	3.35	5.5	0.8	8.32	28.5	-3.1
2380	76	2022-09-22 12:40:00	3.83	6.3	1.7	10.73	27.3	-2.7
2381	77	2022-09-22 12:50:00	3.59	6.1	1.8	10.72	31.1	-0.6
2382	78	2022-09-22 13:00:00	4.11	5.8	2.0	12.85	22.9	4.4
2383	79	2022-09-22 13:10:00	3.8	6.0	1.8	12.82	30.2	2.5
2384	80	2022-09-22 13:20:00	4.29	6.7	1.8	15.69	33.5	1.9
2385	81	2022-09-22 13:30:00	3.89	6.6	1.7	13.32	30.4	1.1
2386	82	2022-09-22 13:40:00	2.78	4.7	1.4	4.39	16.1	-3.4
2387	83	2022-09-22 13:50:00	3.93	6.8	1.0	13.81	34.0	1.4
2388	84	2022-09-22 14:00:00	4.44	7.5	2.1	17.68	47.3	2.3
2389	85	2022-09-22 14:10:00	3.8	7.2	1.9	12.84	37.1	0.5
2390	86	2022-09-22 14:20:00	3.46	8.1	1.0	11.05	46.5	-3.5
2391	87	2022-09-22 14:30:00	4.22	6.6	2.0	14.74	32.0	3.7
2392	88	2022-09-22 14:40:00	3.02	5.8	1.1	5.91	25.6	-3.7
2393	89	2022-09-22 14:50:00	2.74	4.6	1.1	3.77	19.8	-3.8
2394	90	2022-09-22 15:00:00	5.31	8.8	3.0	28.99	64.3	-1.5

FIELD_19_FILL	64-bit integer
FIELD_19_NAME	String, length = 15, padding = H5T_STR_NULLTERM, cset = H5T_CSE
FIELD_1_FILL	String, length = 1, padding = H5T_STR_NULLTERM, cset = H5T_CSE
FIELD_1_NAME	String, length = 9, padding = H5T_STR_NULLTERM, cset = H5T_CSE
FIELD_20_FILL	64-bit integer
FIELD_20_NAME	String, length = 21, padding = H5T_STR_NULLTERM, cset = H5T_CSE
FIELD_21_FILL	64-bit integer

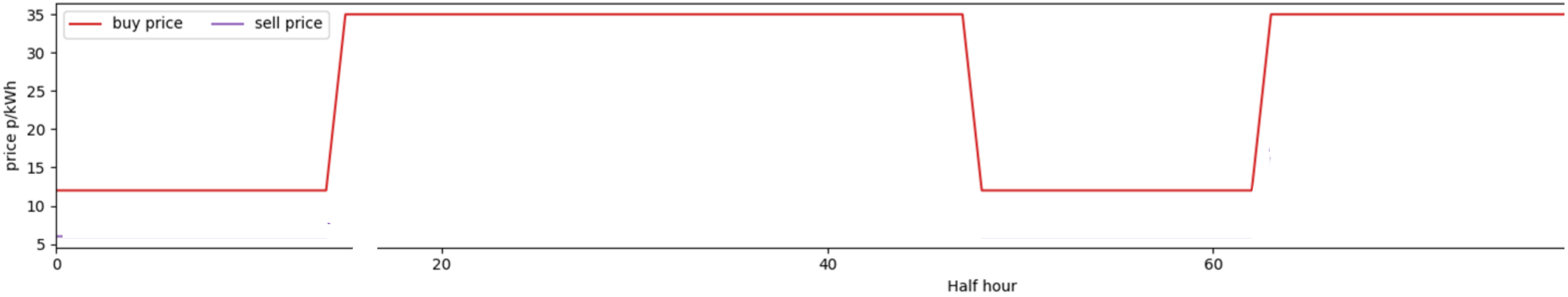


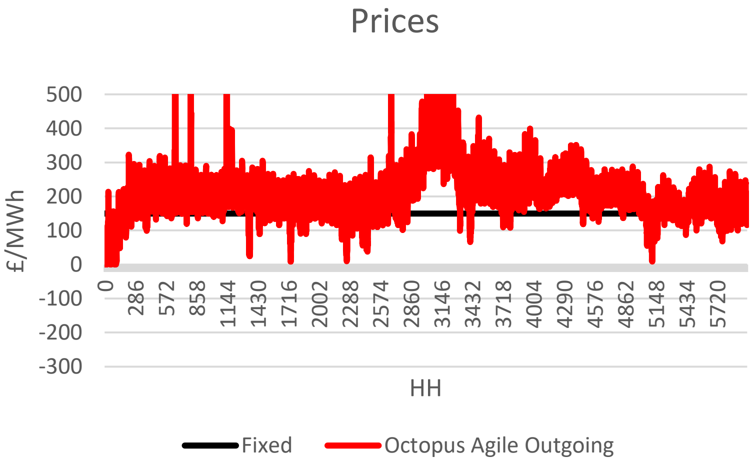


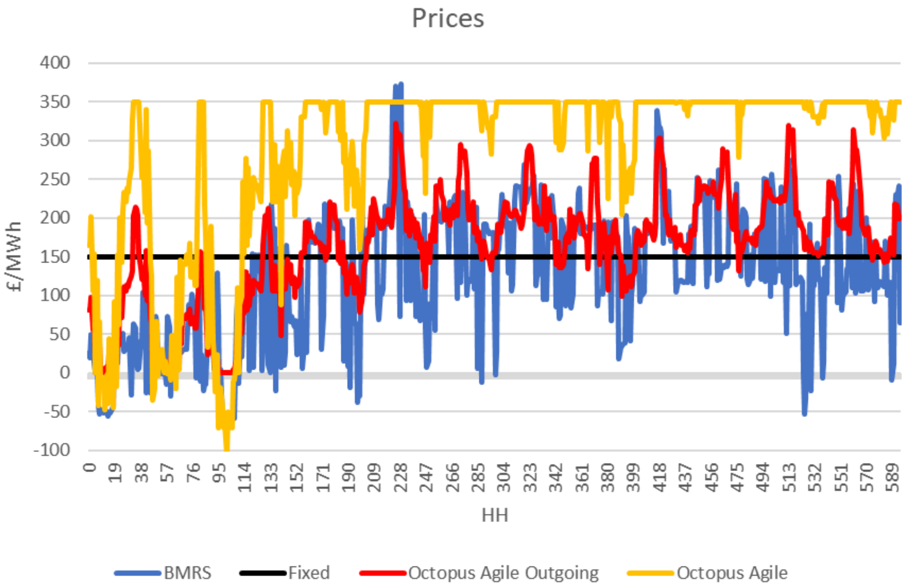


Get Market Prices

2



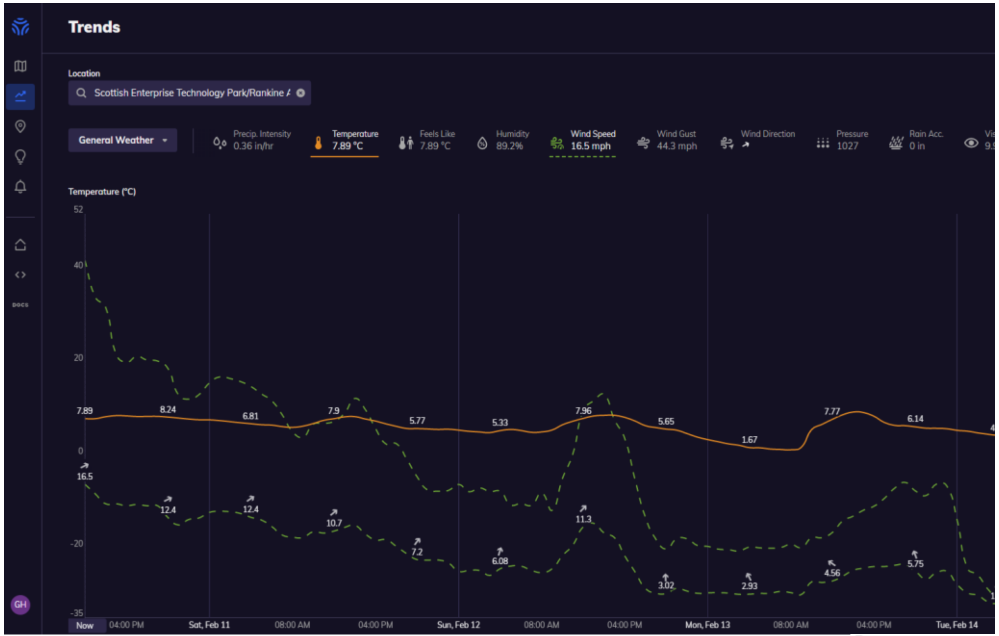




Get Weather Forecasts

3

University of Strathclyde Engineering



- Tomorrow.io
- 4 Days ahead
- Forecasts Stored for later analysis

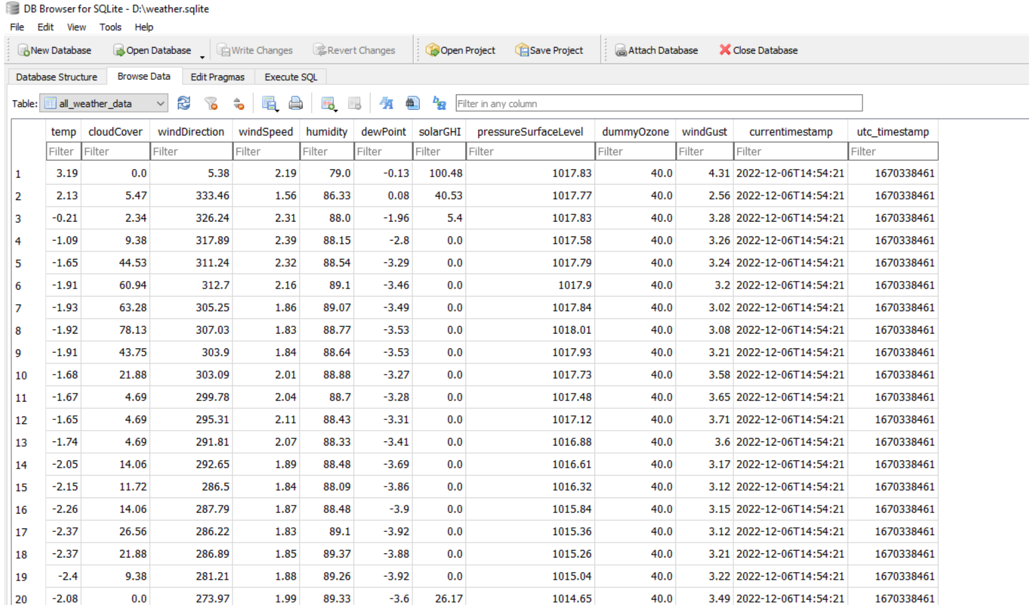
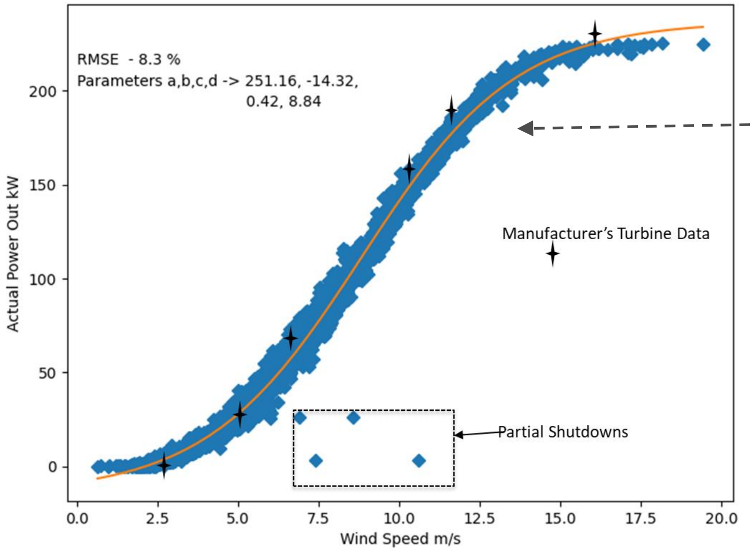


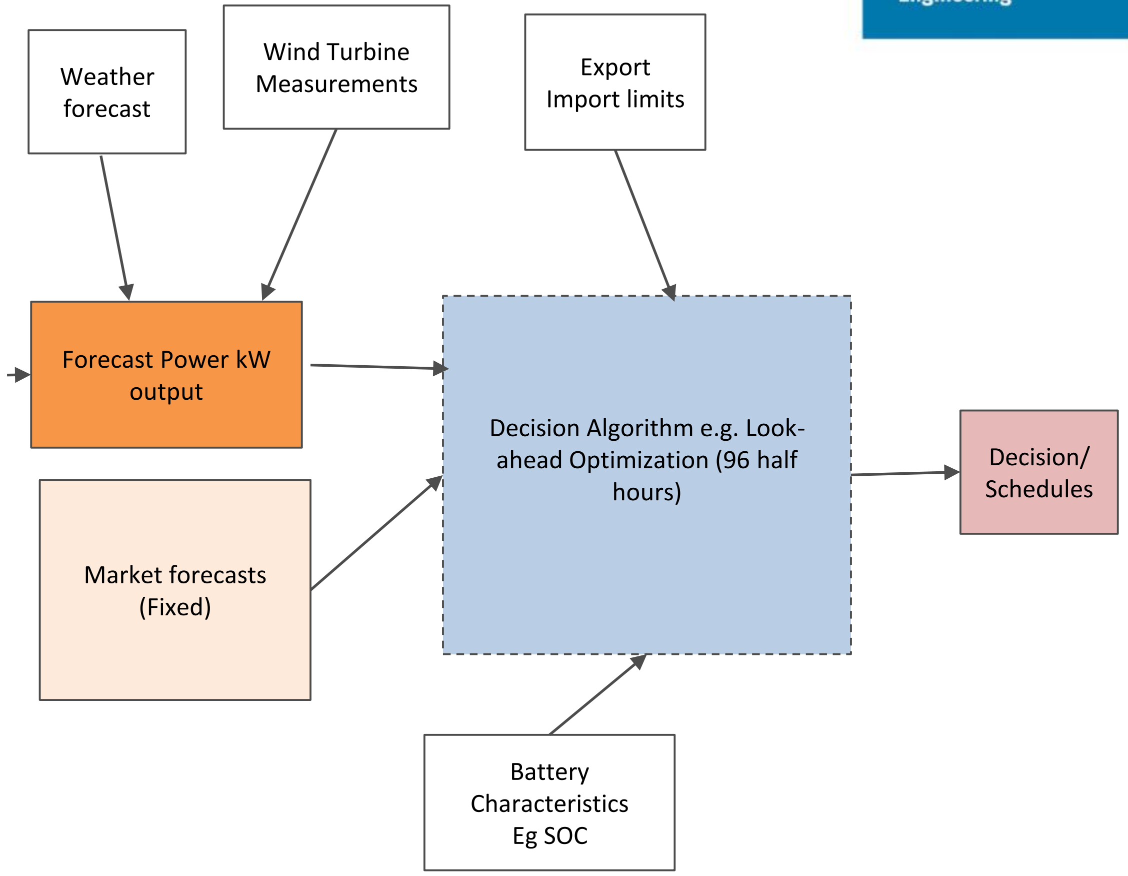
Table: all_weather_data

	temp	cloudCover	windDirection	windSpeed	humidity	dewPoint	solarGHI	pressureSurfaceLevel	dummyOzone	windGust	currentimestamp	utc_timestamp
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	3.19	0.0	5.38	2.19	79.0	-0.13	100.48	1017.83	40.0	4.31	2022-12-06T14:54:21	1670338461
2	2.13	5.47	333.46	1.56	86.33	0.08	40.53	1017.77	40.0	2.56	2022-12-06T14:54:21	1670338461
3	-0.21	2.34	326.24	2.31	88.0	-1.96	5.4	1017.83	40.0	3.28	2022-12-06T14:54:21	1670338461
4	-1.09	9.38	317.89	2.39	88.15	-2.8	0.0	1017.58	40.0	3.26	2022-12-06T14:54:21	1670338461
5	-1.65	44.53	311.24	2.32	88.54	-3.29	0.0	1017.79	40.0	3.24	2022-12-06T14:54:21	1670338461
6	-1.91	60.94	312.7	2.16	89.1	-3.46	0.0	1017.9	40.0	3.2	2022-12-06T14:54:21	1670338461
7	-1.93	63.28	305.25	1.86	89.07	-3.49	0.0	1017.84	40.0	3.02	2022-12-06T14:54:21	1670338461
8	-1.92	78.13	307.03	1.83	88.77	-3.53	0.0	1018.01	40.0	3.08	2022-12-06T14:54:21	1670338461
9	-1.91	43.75	303.9	1.84	88.64	-3.53	0.0	1017.93	40.0	3.21	2022-12-06T14:54:21	1670338461
10	-1.68	21.88	303.09	2.01	88.88	-3.27	0.0	1017.73	40.0	3.58	2022-12-06T14:54:21	1670338461
11	-1.67	4.69	299.78	2.04	88.7	-3.28	0.0	1017.48	40.0	3.65	2022-12-06T14:54:21	1670338461
12	-1.65	4.69	295.31	2.11	88.43	-3.31	0.0	1017.12	40.0	3.71	2022-12-06T14:54:21	1670338461
13	-1.74	4.69	291.81	2.07	88.33	-3.41	0.0	1016.88	40.0	3.6	2022-12-06T14:54:21	1670338461
14	-2.05	14.06	292.65	1.89	88.48	-3.69	0.0	1016.61	40.0	3.17	2022-12-06T14:54:21	1670338461
15	-2.15	11.72	286.5	1.84	88.09	-3.86	0.0	1016.32	40.0	3.12	2022-12-06T14:54:21	1670338461
16	-2.26	14.06	287.79	1.87	88.48	-3.9	0.0	1015.84	40.0	3.15	2022-12-06T14:54:21	1670338461
17	-2.37	26.56	286.22	1.83	89.1	-3.92	0.0	1015.36	40.0	3.12	2022-12-06T14:54:21	1670338461
18	-2.37	21.88	286.89	1.85	89.37	-3.88	0.0	1015.26	40.0	3.21	2022-12-06T14:54:21	1670338461
19	-2.4	9.38	281.21	1.88	89.26	-3.92	0.0	1015.04	40.0	3.22	2022-12-06T14:54:21	1670338461
20	-2.08	0.0	273.97	1.99	89.33	-3.6	26.17	1014.65	40.0	3.49	2022-12-06T14:54:21	1670338461

- Also used in power and heat forecasting models

Forecasts





Weather forecast

Wind Turbine Measurements

Export Import limits

Forecast Power kW output

Market forecasts (Fixed)

Decision Algorithm e.g. Look-ahead Optimization (96 half hours)

Decision/ Schedules

Battery Characteristics Eg SOC

Use forecasts to Maximise Revenues !!!!

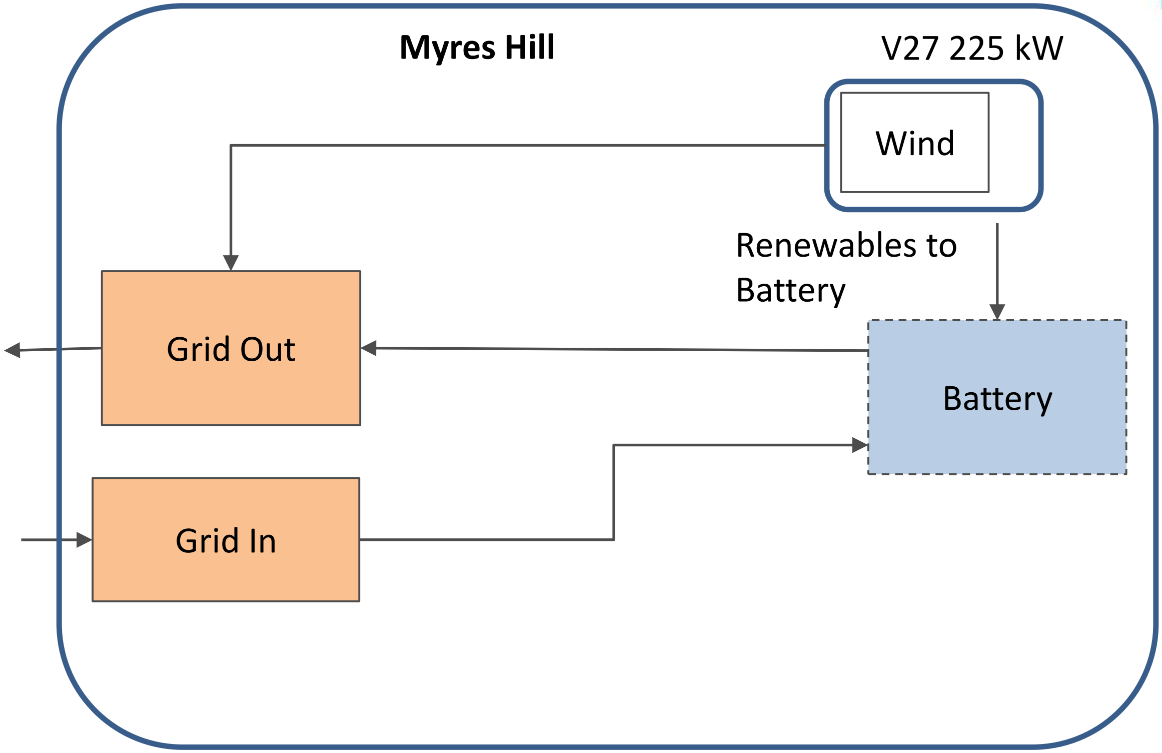
Simple Use Case Decisions

Ideally Maximize Net Revenues

Export Revs – Import Revs

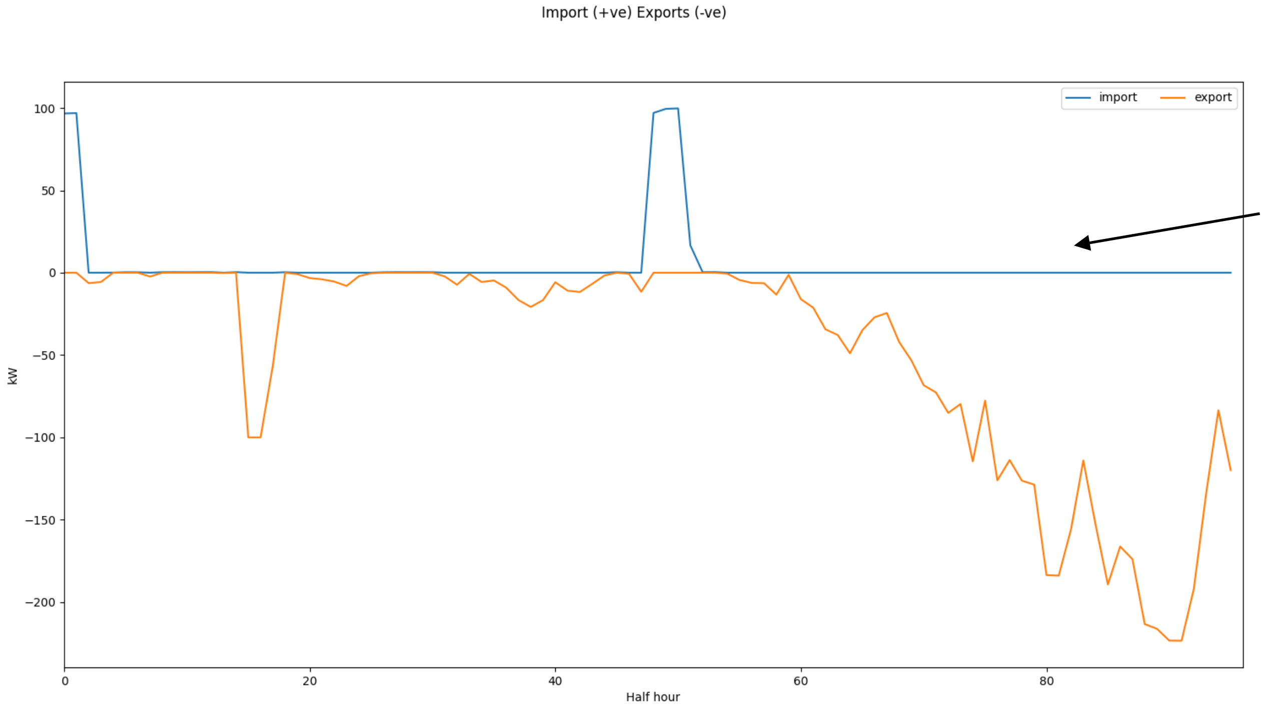
<u>2 cases</u>

- Buy lo sell hi
- Optimize



Imports/Exports

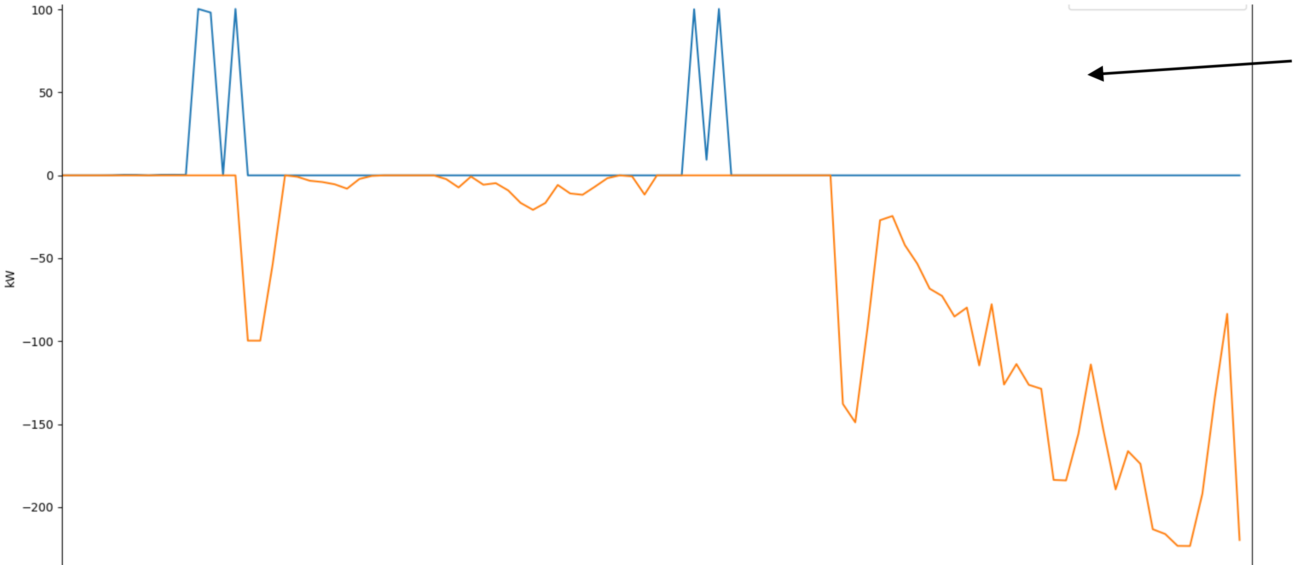
Myres hill simple



Simple strategy

Estimated Revs=£332.5

Renew to batt – 4kWh



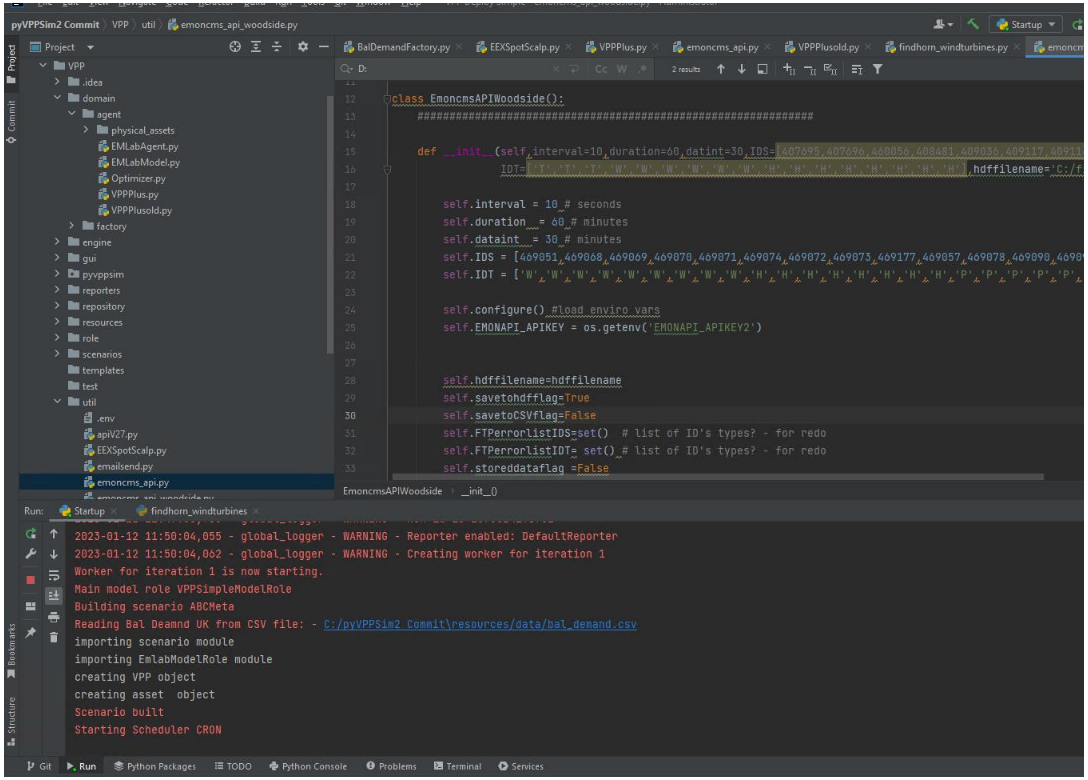
Optimization: <u>Maximise Revenues</u> over 2 days using forecasts

Estimated Revs=£344.3

+3.4%

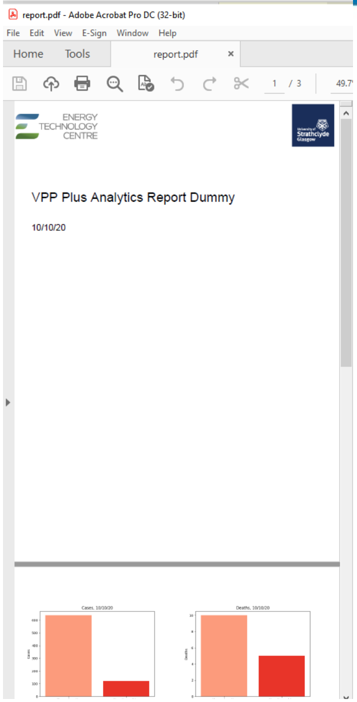
Renew to batt –63kWh

VPP Deployment



```python
class EmoncmsAPIWoodside():
    ####################################################################

    def __init__(self,interval=10,duration=60,datint=30,IDS=[407695,407696,460056,408481,409036,409117,40911
                 IDT=['T','T','T','W','W','W','W','W','W','H','H','H','H','H','H','H','H'],hdffilename='C:/f

        self.interval = 10 # seconds
        self.duration  = 60 # minutes
        self.dataint  = 30 # minutes
        self.IDS = [469051,469068,469069,469070,469071,469074,469072,469073,469177,469057,469078,469090,4690
        self.IDT = ['W','W','W','W','W','W','W','W','W','H','H','H','H','H','H','H','H','P','P','P','P','P',

        self.configure() #load enviro vars
        self.EMONAPI_APIKEY = os.getenv('EMONAPI_APIKEY2')


        self.hdffilename=hdffilename
        self.savetohdfflag=True
        self.savetoCSVflag=False
        self.FTPerrorlistIDS=set()  # list of ID's types? - for redo
        self.FTPerrorlistIDT= set() # list of ID's types? - for redo
        self.storeddataflag =False
```

```
2023-01-12 11:50:04,055 - global_logger - WARNING - Reporter enabled: DefaultReporter
2023-01-12 11:50:04,062 - global_logger - WARNING - Creating worker for iteration 1
Worker for iteration 1 is now starting.
Main model role VPPSimpleModelRole
Building scenario ABCMeta
Reading Bal Deamnd UK from CSV file: - C:/pyVPPSim2_Commit\resources/data/bal_demand.csv
importing scenario module
importing EmlabModelRole module
creating VPP object
creating asset  object
Scenario built
Starting Scheduler CRON
```



VPP Plus Analytics Report Dummy

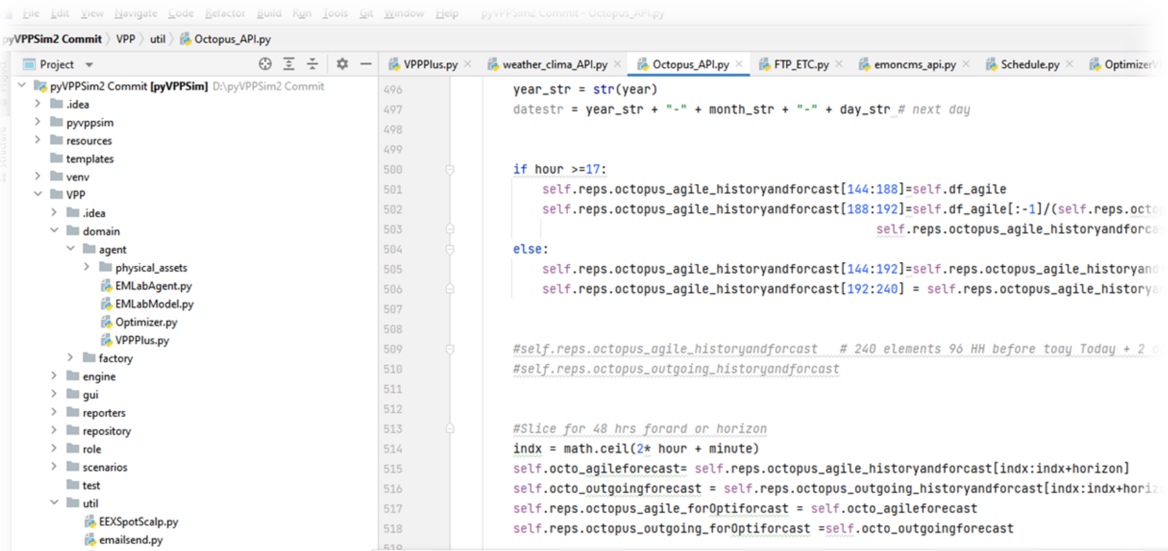
10/10/20

From: testemail@energy-redefined.com
Date: 28 September 2022 at 00:15:52 BST
To: gary.howorth@gmail.com
Subject: error message from VPP Platform

Error on VPP Plus Platform. See Messages/Error Codes below!

errormessage API List >>
+
errormessage Hdf Store List >> ['Octopus 2 Day forecast - Stored Outgoing data Hdf error '] ['EmonCMS Actual Stored data Hdf error ']

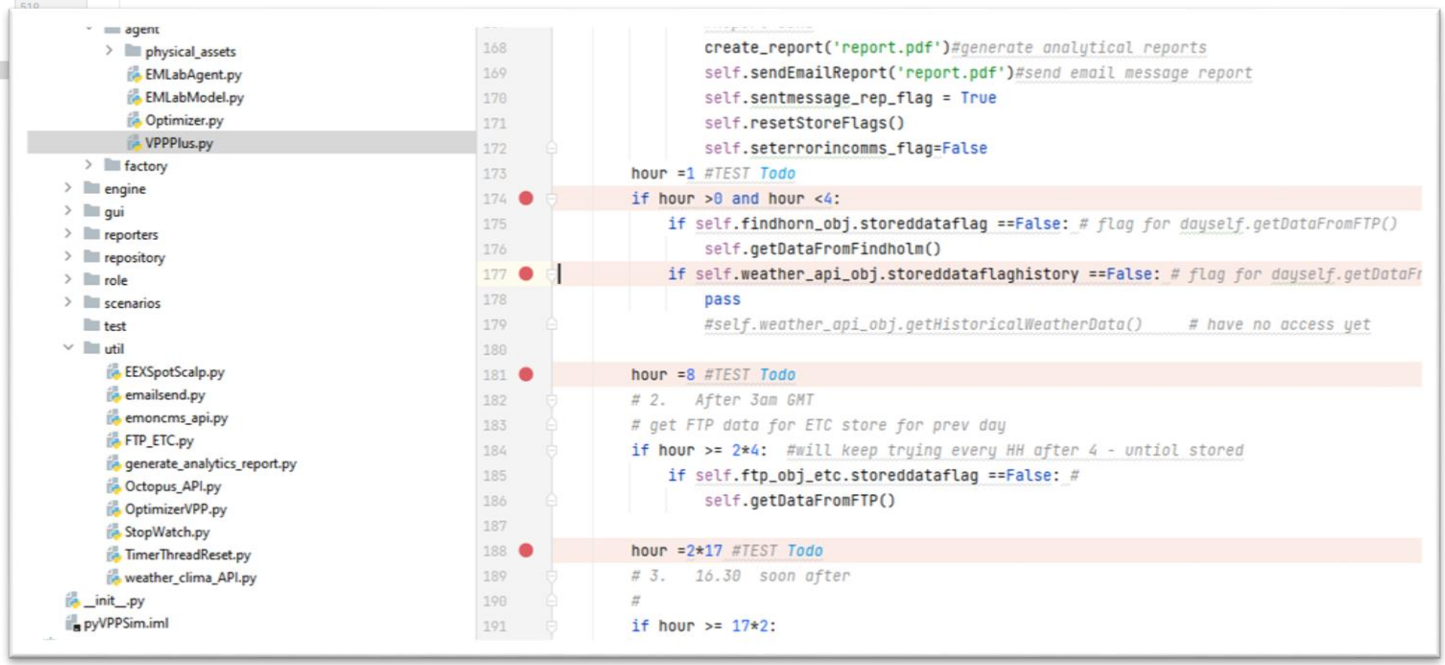
VPP Platform Code-Base


File Edit View Navigate Code Refactor Build Run Tools Git Window Help pyVPPSim2 Commit - Octopus_API.py
pyVPPSim2 Commit 〉 VPP 〉 util 〉 Octopus_API.py
Project
VPPPlus.py × weather_clima_API.py × Octopus_API.py × FTP_ETC.py × emoncms_api.py × Schedule.py × Optimizer

```
pyVPPSim2 Commit [pyVPPSim] D:\pyVPPSim2 Commit
  .idea
  pyvppsim
  resources
  templates
  venv
  VPP
    .idea
    domain
      agent
        physical_assets
        EMLabAgent.py
        EMLabModel.py
        Optimizer.py
        VPPPlus.py
      factory
    engine
    gui
    reporters
    repository
    role
    scenarios
    test
    util
      EEXSpotScalp.py
      emailsend.py
      emoncms_api.py
      FTP_ETC.py
      generate_analytics_report.py
      Octopus_API.py
      OptimizerVPP.py
      StopWatch.py
      TimerThreadReset.py
      weather_clima_API.py
    __init__.py
    pyVPPSim.iml
External Libraries
Scratches and Consoles
```

```python
year_str = str(year)
datestr = year_str + "-" + month_str + "-" + day_str # next day


if hour >=17:
    self.reps.octopus_agile_historyandforcast[144:188]=self.df_agile
    self.reps.octopus_agile_historyandforcast[188:192]=self.df_agile[:-1]/(self.reps.octo
                                              self.reps.octopus_agile_historyandforca
else:
    self.reps.octopus_agile_historyandforcast[144:192]=self.reps.octopus_agile_historyand
    self.reps.octopus_agile_historyandforcast[192:240] = self.reps.octopus_agile_historya


#self.reps.octopus_agile_historyandforcast   # 240 elements 96 HH before toay Today + 2
#self.reps.octopus_outgoing_historyandforcast


#Slice for 48 hrs forard or horizon
indx = math.ceil(2* hour + minute)
self.octo_agileforecast= self.reps.octopus_agile_historyandforcast[indx:indx+horizon]
self.octo_outgoingforecast = self.reps.octopus_outgoing_historyandforcast[indx:indx+hori
self.reps.octopus_agile_forOptiforcast = self.octo_agileforecast
self.reps.octopus_outgoing_forOptiforcast =self.octo_outgoingforecast
```

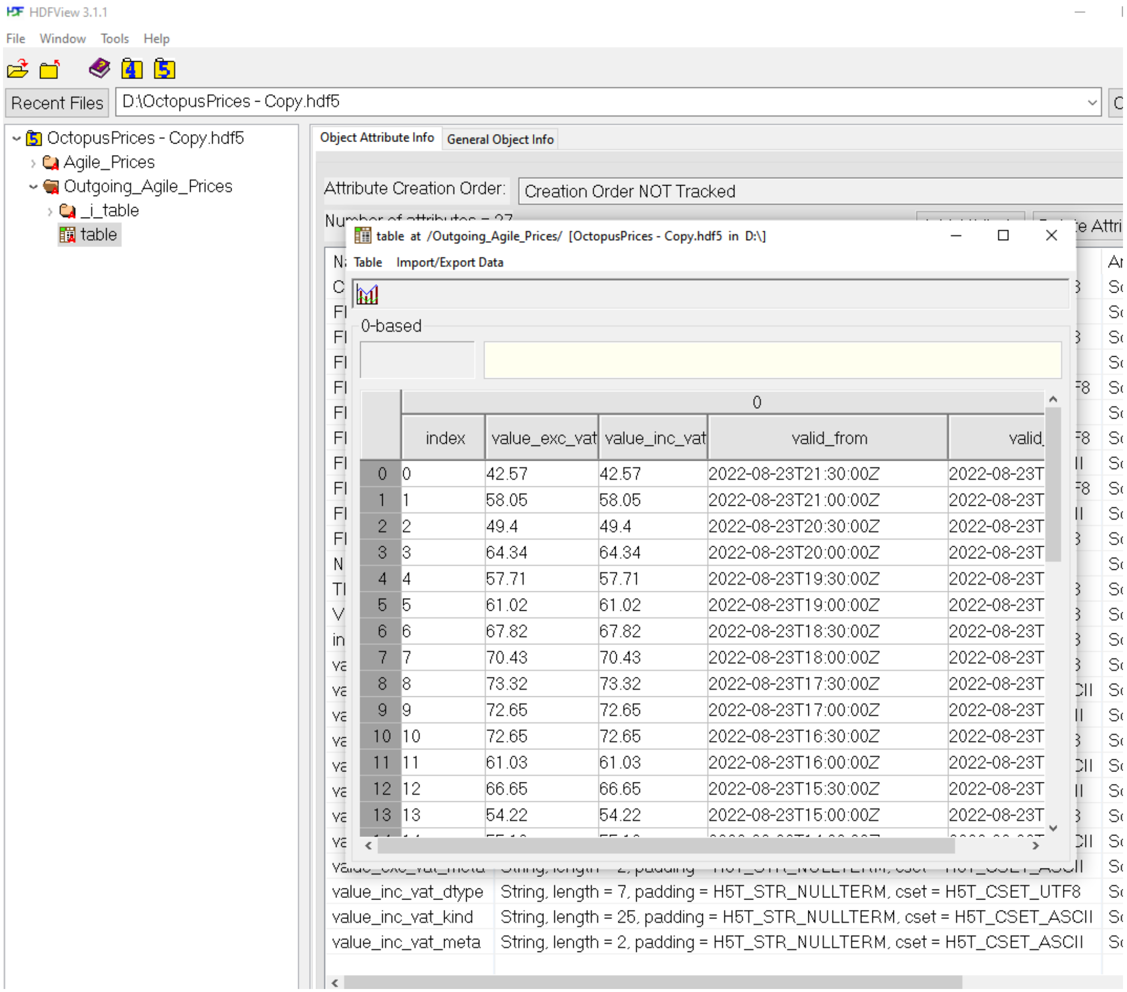
Code based on work by GHoworth 2017-2020
and open source libraries
EmLab 2017



```
  agent
    physical_assets
    EMLabAgent.py
    EMLabModel.py
    Optimizer.py
    VPPPlus.py
  factory
engine
gui
reporters
repository
role
scenarios
test
util
  EEXSpotScalp.py
  emailsend.py
  emoncms_api.py
  FTP_ETC.py
  generate_analytics_report.py
  Octopus_API.py
  OptimizerVPP.py
  StopWatch.py
  TimerThreadReset.py
  weather_clima_API.py
__init__.py
pyVPPSim.iml
```

```python
            create_report('report.pdf')#generate analytical reports
            self.sendEmailReport('report.pdf')#send email message report
            self.sentmessage_rep_flag = True
            self.resetStoreFlags()
            self.seterrorincomms_flag=False
    hour =1 #TEST Todo
    if hour >0 and hour <4:
        if self.findhorn_obj.storeddataflag ==False: # flag for dayself.getDataFromFTP()
            self.getDataFromFindholm()
        if self.weather_api_obj.storeddataflaghistory ==False: # flag for dayself.getDataFr
            pass
            #self.weather_api_obj.getHistoricalWeatherData()     # have no access yet

    hour =8 #TEST Todo
    # 2.   After 3am GMT
    # get FTP data for ETC store for prev day
    if hour >= 2*4:  #will keep trying every HH after 4 - untiol stored
        if self.ftp_obj_etc.storeddataflag ==False: #
            self.getDataFromFTP()

    hour =2*17 #TEST Todo
    # 3.   16.30  soon after
    #
    if hour >= 17*2:
```

Database Example: Hdf5



D:\OctopusPrices - Copy.hdf5

Attribute Creation Order: Creation Order NOT Tracked

table at /Outgoing_Agile_Prices/ [OctopusPrices - Copy.hdf5 in D:\]

	index	value_exc_vat	value_inc_vat	valid_from	valid_
0	0	42.57	42.57	2022-08-23T21:30:00Z	2022-08-23T
1	1	58.05	58.05	2022-08-23T21:00:00Z	2022-08-23T
2	2	49.4	49.4	2022-08-23T20:30:00Z	2022-08-23T
3	3	64.34	64.34	2022-08-23T20:00:00Z	2022-08-23T
4	4	57.71	57.71	2022-08-23T19:30:00Z	2022-08-23T
5	5	61.02	61.02	2022-08-23T19:00:00Z	2022-08-23T
6	6	67.82	67.82	2022-08-23T18:30:00Z	2022-08-23T
7	7	70.43	70.43	2022-08-23T18:00:00Z	2022-08-23T
8	8	73.32	73.32	2022-08-23T17:30:00Z	2022-08-23T
9	9	72.65	72.65	2022-08-23T17:00:00Z	2022-08-23T
10	10	72.65	72.65	2022-08-23T16:30:00Z	2022-08-23T
11	11	61.03	61.03	2022-08-23T16:00:00Z	2022-08-23T
12	12	66.65	66.65	2022-08-23T15:30:00Z	2022-08-23T
13	13	54.22	54.22	2022-08-23T15:00:00Z	2022-08-23T

value_inc_vat_dtype	String, length = 7, padding = H5T_STR_NULLTERM, cset = H5T_CSET_UTF8
value_inc_vat_kind	String, length = 25, padding = H5T_STR_NULLTERM, cset = H5T_CSET_ASCII
value_inc_vat_meta	String, length = 2, padding = H5T_STR_NULLTERM, cset = H5T_CSET_ASCII

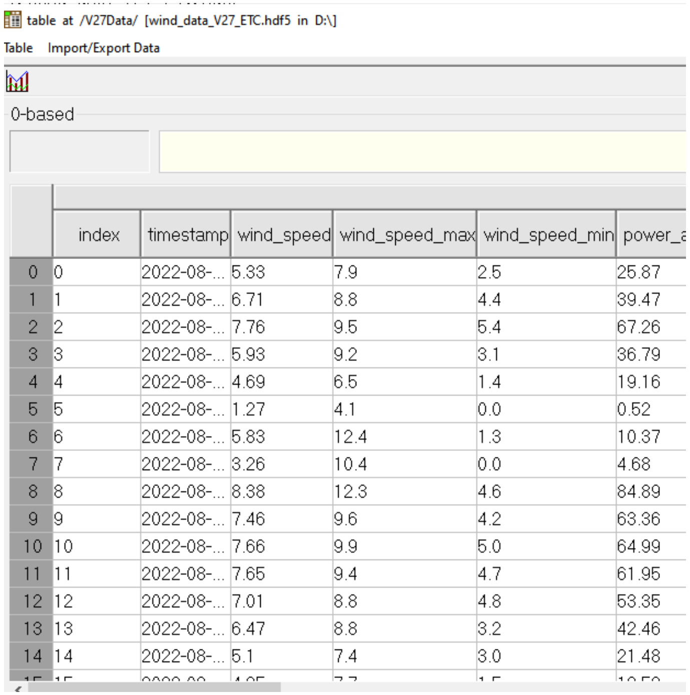


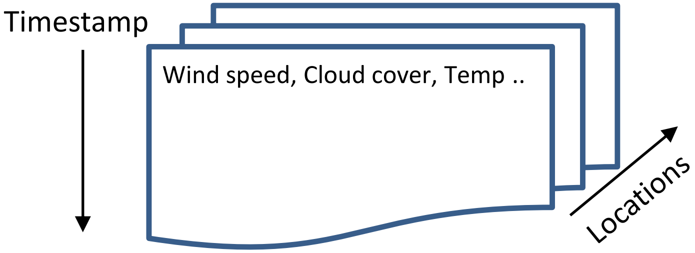
table at /V27Data/ [wind_data_V27_ETC.hdf5 in D:\]

	index	timestamp	wind_speed	wind_speed_max	wind_speed_min	power_a
0	0	2022-08-...	5.33	7.9	2.5	25.87
1	1	2022-08-...	6.71	8.8	4.4	39.47
2	2	2022-08-...	7.76	9.5	5.4	67.26
3	3	2022-08-...	5.93	9.2	3.1	36.79
4	4	2022-08-...	4.69	6.5	1.4	19.16
5	5	2022-08-...	1.27	4.1	0.0	0.52
6	6	2022-08-...	5.83	12.4	1.3	10.37
7	7	2022-08-...	3.26	10.4	0.0	4.68
8	8	2022-08-...	8.38	12.3	4.6	84.89
9	9	2022-08-...	7.46	9.6	4.2	63.36
10	10	2022-08-...	7.66	9.9	5.0	64.99
11	11	2022-08-...	7.65	9.4	4.7	61.95
12	12	2022-08-...	7.01	8.8	4.8	53.35
13	13	2022-08-...	6.47	8.8	3.2	42.46
14	14	2022-08-...	5.1	7.4	3.0	21.48

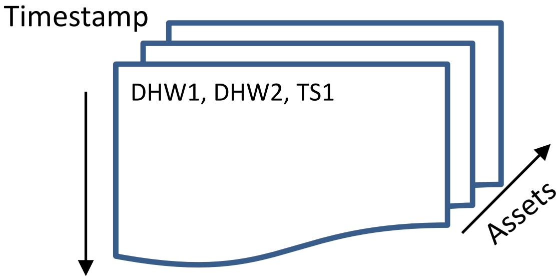
Databases:

University of Strathclyde Engineering

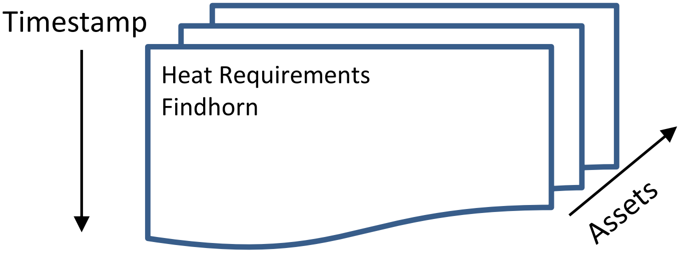
Weather



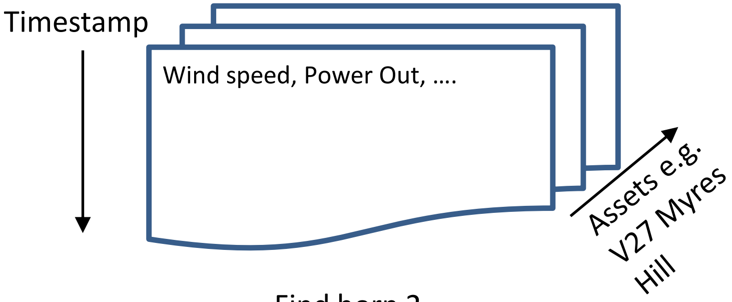
FindHorn 1



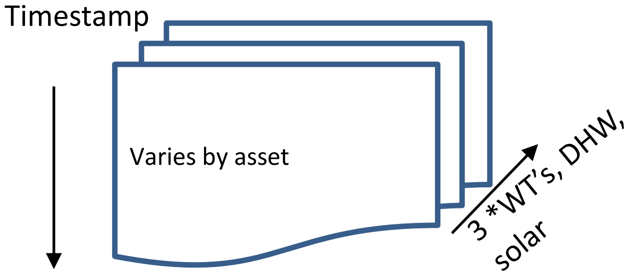
Heat Forecasts



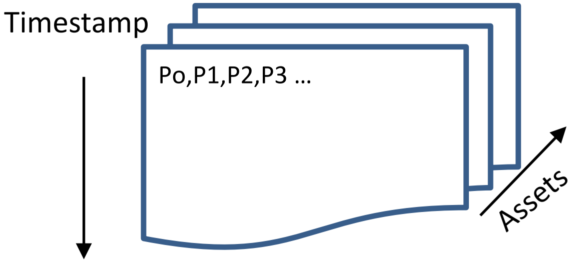
Asset Data Power



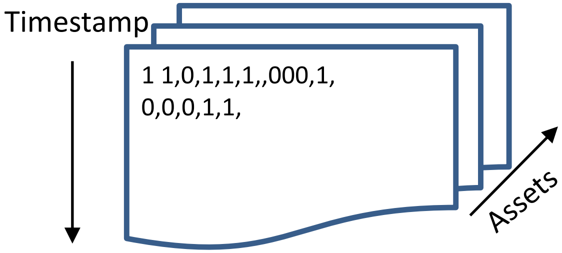
Find horn 2



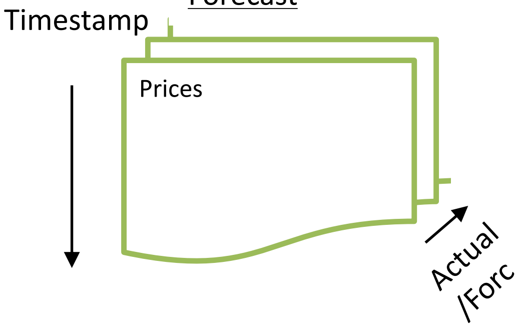
Power Out forecasts



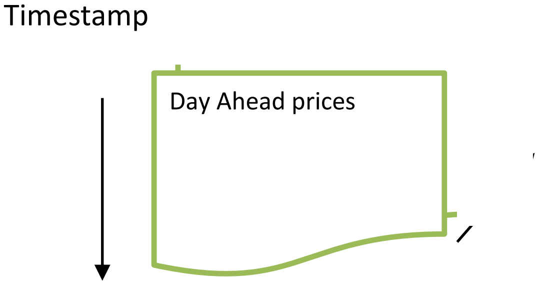
Schedules



Octopus Actual - Forecast

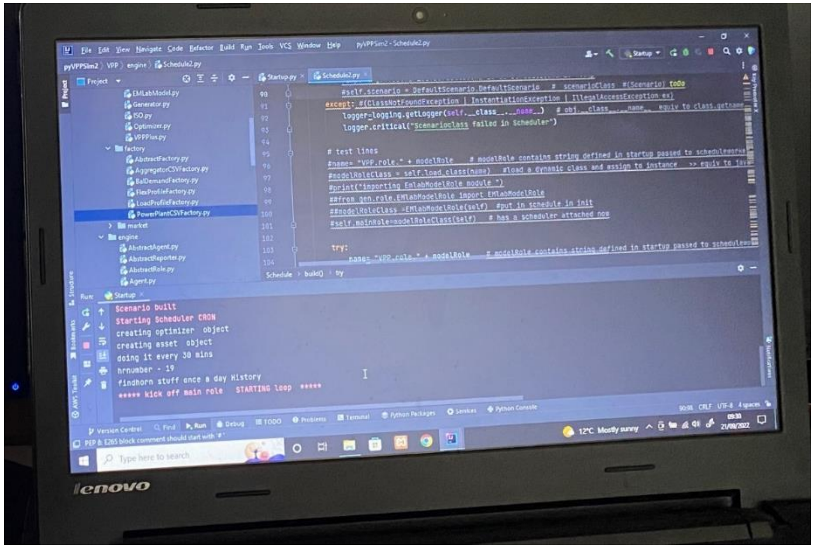


EPEX Actual



Two Versions

- The PyEMLab framework allows us to either simulate VPP actions or use a real time scheduler or both.
- This allows us to perform Hardware in the loop simulations and to simulate future markets that may be evolving and currently do not exist.
- It has been used to test out the effects of different business models and values
- Future Assets can be represented as Digital Twins



```
        #self.scenario = DefaultScenario.DefaultScenario   #  scenarioClass  #(Scenario) toDo
    except: #(ClassNotFoundException | InstantiationException | IllegalAccessException ex)
        logger=logging.getLogger(self.__class__.__name__)   # obj.__class__.__name__  equiv to class.getname
        logger.critical("Scenarioclass failed in Scheduler")

    # test lines
    #name= "VPP.role." + modelRole    # modelRole contains string defined in startup passed to scheduleworke
    #modelRoleClass = self.load_class(name)   #load a dynamic class and assign to instance   >> equiv to jav
    #print("importing EmlabModelRole module ")
    ##from gen.role.EMlabModelRole import EMlabModelRole
    ##modelRoleClass =EMlabModelRole(self)  #put in schedule in init
    #self.mainRole=modelRoleClass(self)   # has a scheduler attached now

    try:
        name= "VPP.role." + modelRole    # modelRole contains string defined in startup passed to schedulee
```

```
Scenario built
Starting Scheduler CRON
creating optimizer  object
creating asset  object
doing it every 30 mins
hrnumber - 19
findhorn stuff once a day History
***** kick off main role   STARTING loop  *****
```

PEP 8: E265 block comment should start with '# '

Lessons Learned

- After an extensive literature review and more detailed discussions, it has become clear that none of the current VPP providers are able to meet all the needs of the SIES project. An off the shelf solution with cloud resources, cyber security and error capturing functionality may have proved to be a better approach, although it will still have necessitated the development and integration of specialized algorithms and routines.
- Furthermore, integrating legacy assets or assets from different manufacturers has been time consuming.
- Prior work using the PyEMLab-AGG framework has allowed us to follow a rapid prototype development process. The rapid prototyping process includes three steps; prototyping, testing, and refining (learning by doing approach). Prior experience tells us that significant software savings can be achieved with this approach as software functionality specification can be improved. Other lessons learned include:

Lessons Learned

- Uncertainty in market price and generation from renewable resources, such as wind, is relatively large. Probabilistic approaches are required, which is driving VPP owners towards using risk management techniques to protect against potential downsides.
- As the number of types of assets and their numbers grow, complexity of models increases, resulting in longer optimization solution times. In the case where the problem is inherently non-linear, very long solution times can occur e.g. > three hours. This can be remedied by linearization and assumption simplifications, but it is important to understand how these can influence accuracy of obtained solutions.
- Pattern recognition matching could be used to help reducing solution times, or when optimizers do not converge.
- Pre-processing of data to exclude spurious scenarios helps improve forecasting, as blindly taking all data can result in worse forecasting models. This is difficult to spot in an online automated models

References

[1] G. Howorth, I. Kockar, P. Tuohy, and J. Bingham, "An enhanced virtual power plant for flexibility services into a local area (including EV's)," in CIRED Porto Workshop 2022: E-mobility and power distribution systems, 2022, vol. 2022, pp. 970-974.

[2] S. You, C. Træholt, and B. Poulsen, "Generic Virtual Power Plants: Management of distributed energy resources under liberalized electricity market," in 8th International Conference on Advances in Power System Control, Operation and Management (APSCOM 2009), 2009, pp. 1-6.

[3] G. Howorth, "Extending the AgentSpring/EMLab Tool to Evaluate Additional Agent Behaviour such as Electric Vehicles and Demand Side Response," ed. ETP Annual Conference 2019 - Energy Technology Partnership Dundee UK: ETP, 2019.

[4] L. J. De Vries, É. J. L. Chappin, and J. C. Richstein, "EMLab-Generation An experimentation environment for electricity policy analysis," 2013.

[5] R. Peters, "cron," Expert Shell Scripting, pp. 81-85, 2009.

[6] B. L. S. Aigner, "System modeling and dispatch schedule optimization of combined PV battery system using linear optimization," Masters, University of Agder, 2021.

References

[7] E. Barbour and M. C. González, "Projecting battery adoption in the prosumer era," Applied energy, vol. 215, pp. 356-370, 2018.

[8] M. L. Bynum et al., Pyomo-optimization modeling in python. Springer, 2021.

[9] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: gradient boosting with categorical features support," arXiv preprint arXiv:1810.11363, 2018.

[10] O. Steinki and Z. Mohammad, "Introduction to ensemble learning," Available at SSRN 2634092, 2015.

[11] A. Lyden, G. Flett, and P. G. Tuohy, "PyLESA: A Python modelling tool for planning-level Local, integrated, and smart Energy Systems Analysis," SoftwareX, vol. 14, p. 100699, 2021/06/01/ 2021.

[12] L. D. Beal, D. C. Hill, R. A. Martin, and J. D. Hedengren, "Gekko optimization suite," Processes, vol. 6, no. 8, p. 106, 2018.

Thank you

University of
Strathclyde
Glasgow