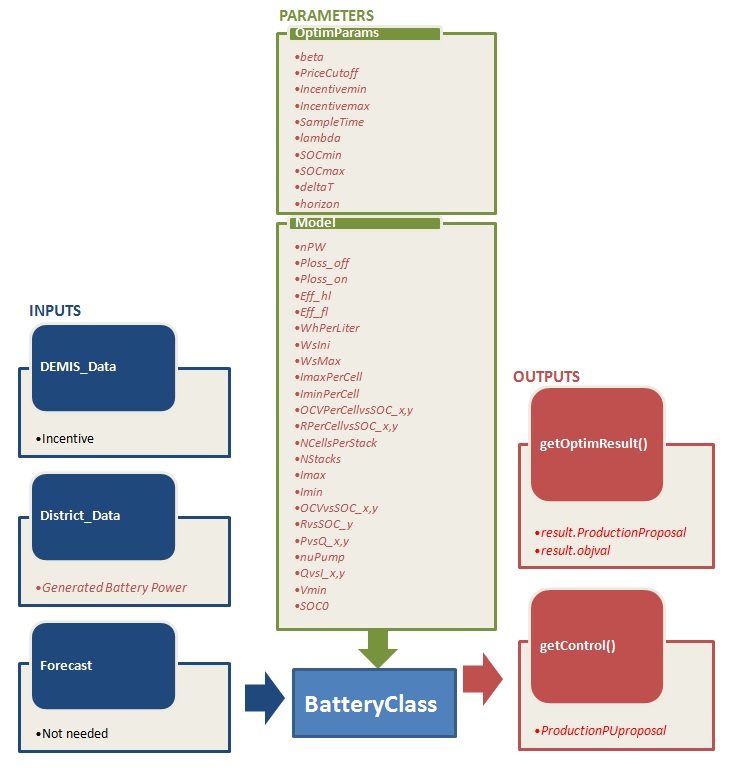
1. eNodes algorithms description
   1. Component Name [e.g. BatteryClassObject]
      1. xMS overview
         1. Diagram



* + - 1. Details on inputs/outputs/parameters

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Description | Unit |

|  |  |  |  |
| --- | --- | --- | --- |
| Inputs | |  |  |
| DEMIS\_Data | * Incentive | For us it is a 96 values vector | -- |
| District\_Data | * *Generated Power* | *Effective generated power (96 values vector)* | *W* |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Forecast | * *Not Needed* |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Outputs | |  |  |
| getOptimResult() | * result. ProductionProposal | Power profile over next horizon (96 15 min time slot power production proposal) | W |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| getControl() | * *ProductionPUproposal* | *Power setpoint sent to power stage in pu* | *pu* |
|  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameters | |  | | |  |
| optimParameters | * IncentiveFactor.beta | Incentive Factor function’s sharpe parameter | | | -- |
|  | * IncentiveFactor.PriceCutoff | Incentive Factor function’s Price Cut off values at Wind Turbine production falls. | | | -- |
|  | * IncentiveFactor.Incentivemin * IncentiveFactor.Incentivemax | Incentive Factor functions’ shape factors | | | -- |
|  | * *SampleTime* * *Lambda* * *SOCmin* * *SOCmax* * *deltaT* * *horizon* | Sample time for power production vector  Ponderation factor for power production or consummation  Minimum SOC level  Maximum SOC level  Battery Dynamical model simulation time step  Optimization time horizon | | | -- |
| Model |  |  | | |  |
|  | * *Param.nPW Battery’s Rated Power Watts* * *Param.Eff.Ploss\_off Power losses in off mode Watts* * *Param.Eff.Ploss\_on Power losses in on mode Watts* * *Param.Eff.Eff\_hl Efficiency in half charge* * *Param.Eff.Eff\_fl Efficiency in full charge* * *Param.Nliters Battery’s flow volume Liters* * *Param.WhPerLiter Energy per liter Wh/L* * *Param.WsIni Initial battery energy Joules* * *Param.WsMax Maximum energy capacity Joules* * *Param.ImaxPerCell Maximum current per cell Amps* * *Param.IminPerCell Minimum current per cell Amps* * *OCVPerCellvsSOC\_x OCV vs SOC Curve’s abscissa pu* * *OCVPerCellvsSOC\_y OCV vs SOC Curves’s ordinate Volts* * *RPerCellvsSOC\_x Resistance curve’s abscissa pu* * *RPerCellvsSOC\_y Resistance curve’s ordinate Ohms* * *NCellsPerStack Number of cells per stack* * *NStacks Number of stacks* * *Imax Maximum current Amps* * *Imin Minimum current Amps* * *OCVvsSOC\_x Open Circuit Voltage curve’s abscissa pu* * *OCVvsSOC\_y Open Circuit Voltage curve’s ordinate Volts* * *RvsSOC\_x Resistance curve’s abscissa pu* * *RvsSOC\_y Resistance curve’s ordinate Ohms* * *PvsQ\_x Power losses curve’s m3/seg* * *PvsQ\_y Power losses curve’s Watts* * *nuPump Pump performance pu* * *QvsI\_x Flow curve’s abscissa Amps* * *QvsI\_y Flow curve’s ordinate m3/seg* * *Vmin Minimum voltage value to avoid zero division error* * *SOC0 Initial State of Charge* |  | | |  |
|  |  |  |  | | |
|  |  |  | |  | |

* + 1. Optimization problem generation
       1. Incentive factor concept and the cost function analisys

The Incentive factor that depends on DEMIS\_Data’s Incentive function. The incentive factor can be defined in the next equations:

The incentive function lets to DEMIS to modulate how much energy wants to capture from the battery.

Our cost function can be defined as follows:

That is subject to the battery dynamic:

We suppose that f(E) is equal to zero, because the energy losses are negliage.

So finally the power setpoint to the power converter can be expressed as follows:

* + - 1. Embedded component model

The battery model is defined as a dynamical model. The dynamical model presents an algebraic loop.

(eq1.)

This model has an algebraic loop created by the currents and the power losses, SOC and OCV. This loop is broken in the numerical integration loop.

* + - 1. Limits

Assumputions:

When we have proposed the power dispatch algorithm we have supposed that the energy lossed stored in the battery is negliable. But this losses haved been taken into account in the dynamical behaviour of the battery. Finally, we have used a variational calculus to have a optimal solution that can be calculated very quickly because a genetic or intelligent algorithm approach takes too much computational time.

* + 1. Optimization problem resolution

*The optimization algorithm is fully integrared in the model using the eq1. The optimization algorithm is based on variational calculus theory in order to have a fast energy dispatch algorithm (see eq1.).*

* + 1. Algorithm results illustration

*We show a optimization with a price wave that generates a Incentive factor with a “sinus” shape. The incentive factors function is defined in fig2.*

*IF.emf*

*Fig1*

*Incentive factor function plot with Price Cutoff=0.9, beta=0.85, Incentivemin=0, Incentivemax=1.*

*IFvsPrice.emf*

*Fig2*

*In the next figure, we show the real power production in red, and the proposed power production in green.*

*Power.emf*

*Fig3*

*SOC.emf*

*Fig4*

*The State of Charge (SOC) is shown in Fig4. In same times the SOC arrives to the SOC limits. In this cases the control charges or discharges and the optimization law is not followed (See fig).*

*TwoCurves.emf*

*Fig5*