DOCUMENTATION OF THE ECOGRID EU PHASE 2 MARKET IMPLEMENTATION

Authors: Bernhard Jansen bja@zurich.ibm.com
Fabian Mueller fmu@zurich.ibm.com

Affiliation: IBM Research Zurich

Version: 1.0
Last mod.: November 16, 2014

1 SCOPE

This document describes the implementation of the EcoGrid phase 2 market. Section 2 provides an overview of the main components of the market, their interrelatedness and the data they rely on. Section 3 is dedicated to a discussion of the timing and synchronization of the different steps involved in the real-time price market. The market initialization procedure is documented in Section 4 and the market data persistency is covered in Section 5.

2 OVERVIEW OF MARKET COMPONENTS

The EcoGrid phase 2 market computes a 5 minute real-time price given a measure of the imbalance of the grid. The computation of the 5min price updates involves the consecutive execution of different interdependent steps that can be divided into two groups: the load forecasting module and the real-time price engine. Figure 1 provides an overview of the different components that make up the phase 2 market. The load forecasting module consists of the components colored in light blue, whereas the real-time price engine consists of the red components.

2.1 LOAD FORECASTING MODULE

The load forecasting module consists of an online and an offline component. The offline component identifies the parameters of a time series model predicting the aggregate power consumption of a group of loads. The parameter identification relies on historic measurement data of the aggregate energy consumption, the spot price, and the outdoor temperature. One of the parameters identified is the sensitivity of the aggregate load with regard to changes in the committed price. The parameters are used in the load forecasting model that is run online in the market. The states of the model are updated every 5 minutes taking into account the latest aggregate load measurement, the previous committed 5 minute real-time price, and the current outdoor temperature. Every 30 minutes, the load forecasting model computes a load forecast for the next 3 hours based on forecasted outdoor temperature and day-ahead prices.

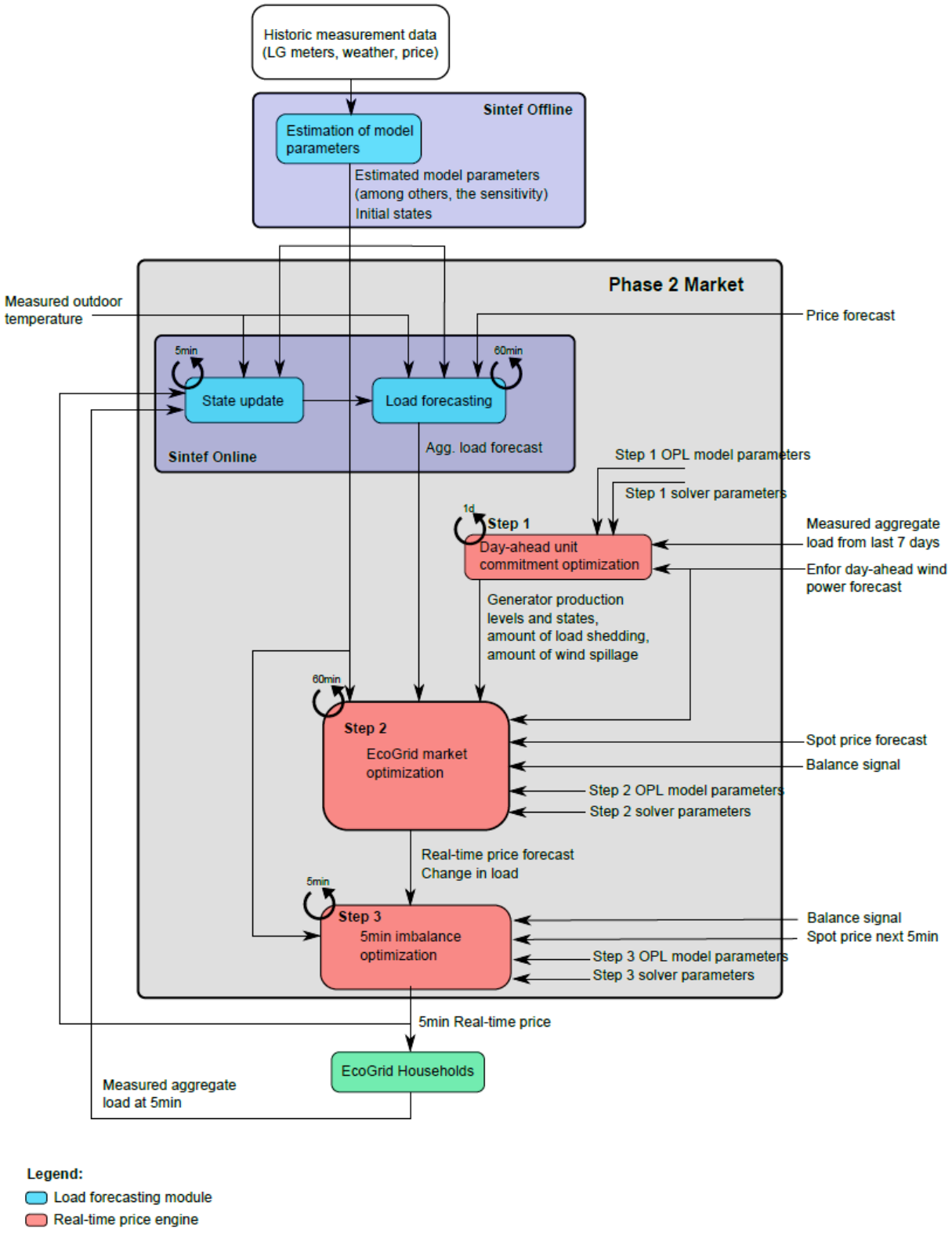

Figure 1 Overview of the different phase 2 market components and their interrelatedness.

2.1.1 STATE UPDATE COMPONENT

The state update component is executed every 5 minutes to update the states of the load forecasting model. Its inputs and outputs are summarized in Table 1 below.

Variable name	Type	Dimension	Source
Model parameters	double	array	Offline parameters, see documentation of offline parameter identification by Sintef.
Measured aggregate load	double	scalar	The latest available aggregated load of at least 1000 meters scaled to 1900 meters.
Committed 5min price	double	scalar	Output of Step 3 OPL
Outdoor temperature	double	scalar	The outdoor temperature forecast from DMI for the latest available meter data.

Table 1 Inputs to the state update module.

2.1.1.1 COMPUTATION OF THE LONG-TERM TREND DATA INDEX

The state update of the load forecasting model requires that particular elements from long-term trend consumption vectors are used. The individual indices of those elements depend on the time interval of the measurement data that was used for identification and on the current time.

The long-term data included in the offline parameter set 'wrkspace_18feb2014_all.mat' does not cover an entire year which makes it necessary to change the way of computing the correct indices. In order to be able to work with the limited amount of data available, monthly and weekly trends are neglected. The long-term trend indices are computed as follows:

Input: Starting time of offline identification: DD, HH:MM:SS

1. t = number of seconds since last DD, HH:MM:SS until now
2. idx_t = floor(tc/(60*5));

Output: idx_t

2.1.2 LOAD FORECAST COMPONENT

Given the current state, the load forecasting component computes a load forecast for the next 3 hours. The input and output variables are summarized in Tables 2 and 3.

Variable name	Type	Dimension	Description
Model parameters	double	array	Offline parameters, see documentation of offline parameter identification by Sintef.
Current model states	double	array	Output of the state update component.
Outdoor temperature forecast	double	array	The outdoor temperature forecast from DMI
Price forecast	double	array	Day-ahead price

Table 2 Inputs to the load forecasting component.

Variable name	Type	Dimension	Description
Load forecast	double	array	Forecast of the aggregate load for the next 3 hours.

Table 3 Output of the load forecasting component.

2.2 REAL-TIME PRICE ENGINE

The real-time price engine comprises three steps:

1. The day-ahead unit commitment step computes the optimal generator production levels, their operation states, as well as the required amount of load shedding and wind spillage.
2. The EcoGrid hourly market optimization step computes the 5 minute real-time price for the next 3 hours.
3. The 5 minute imbalance optimization computes the actual 5 minute commit price with the aim of compensating for the current grid imbalance.

2.2.1 DAY-AHEAD UNIT COMMITMENT OPTIMIZATION

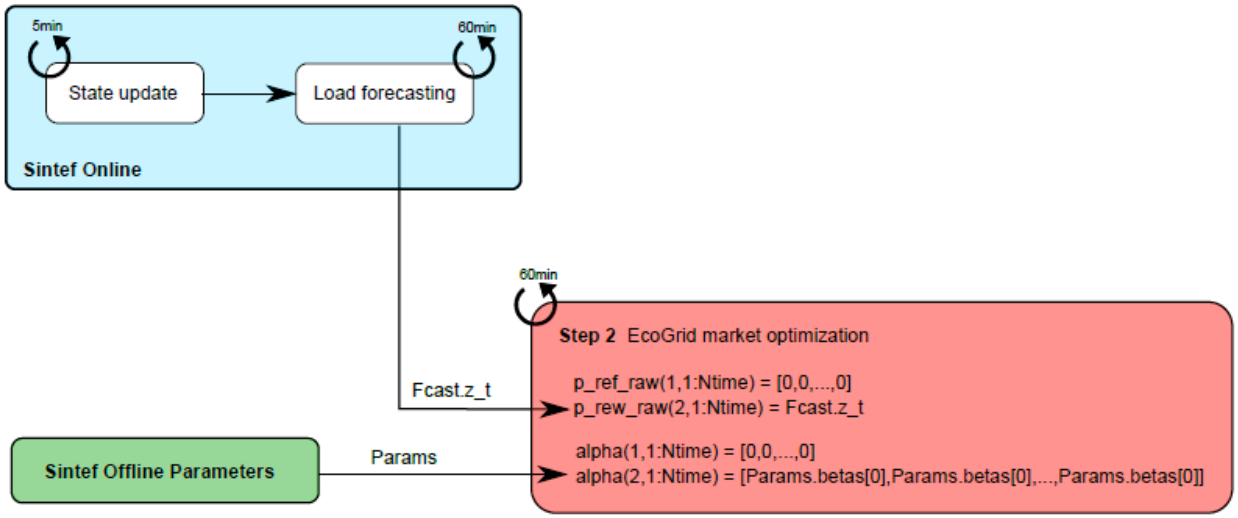

Figure 2 Details of how the load forecasting component and step 2 of the real-price engine interact.

Variable name	Type	Dimension	Description
Step 1 parameters	double	array	Fixed parameters for the step 1 OPL
week_load	double	array	The 95%-quantile of the load from the last 7 days
p_da_wind	double	array	The day-ahead wind power forecast by Enfor

Table 4 Inputs to step 1 of the real-price engine.

Variable name	Type	Dimension	Description
p_sch	double	array	Production levels of the generators
p_da_load_shed	double	array	Amount of load shedding
p_wind_spill	double	array	Amount of wind spillage
u	Boolean	array	Generator state (ON/OFF)
q_max	double	array	Generator scaling factors

Table 5 Outputs of step 1 of the real-price engine.

2.2.2 HOURLY MARKET OPTIMIZATION

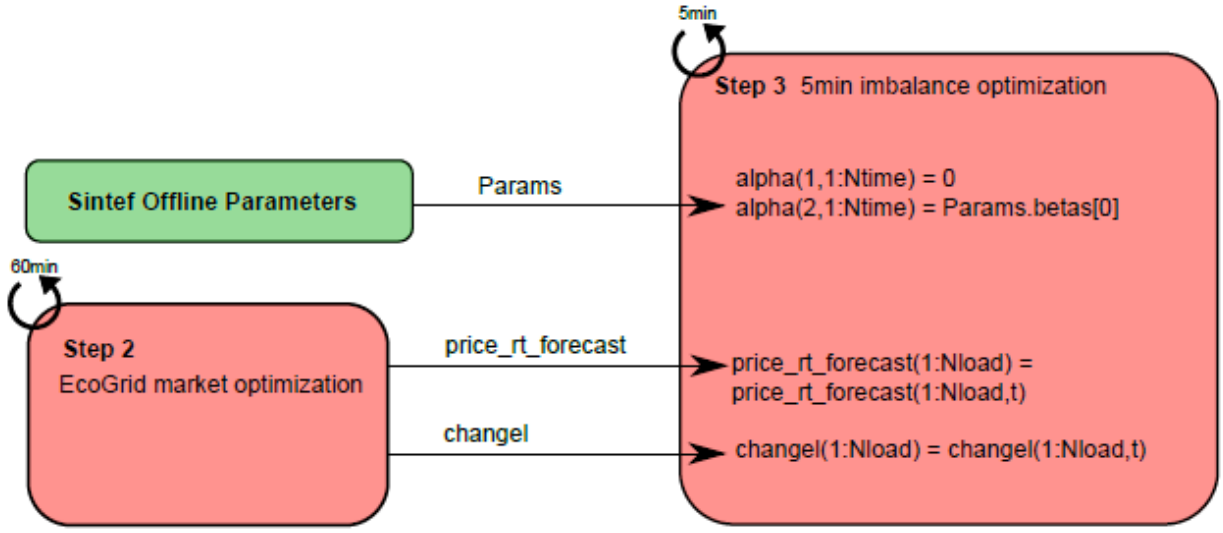

Figure 3 Details of the interaction between step 2 and step 3 of the real-price engine.

Variable name	Type	Dimension	Description
Step 2 parameters	double	array	Fixed parameters for the step 2 OPL
p_sch	double	array	Output from step 1
q_max	double	array	Output from step 1
p_ref_raw	double	array	Load forecast from the load forecasting module
spot_price	double	array	Spot price
alpha	double	scalar	Load sensitivity from Sintef offline parameter identification
p_da_wind	double	array	Day-ahead wind forecast from Enfor
balance	double	array	Expected grid imbalance. Currently, the expected wind power forecast error is used as a measure of imbalance.

Table 6 Inputs to step 2 of the real-price engine.

Variable name	Type	Dimension	Description
price_rt_forecast	double	array	Real-time price forecast for the next 3 hours
changel	double	array	Change in load

Table 7 Outputs of step 2 of the real-price engine.

2.2.3 5 MIN IMBALANCE OPTIMIZATION

Variable name	Type	Dimension	Description
Step 3 parameters	double	array	Fixed parameters for the step 3 OPL
changel	double	array	Output from step 2
price_rt_forecast	double	array	Real-time price from step 2
alpha	double	scalar	Load sensitivity from Sintef offline parameters

Table 8 Inputs to step 3 of the real-price engine.

Variable name	Type	Dimension	Description
price_rt	double	scalar	5min commit price

Table 9 Outputs of step 3 of the real-price engine.

3 TIMING OF MARKET EVENTS

As discussed in Section 2, the phase 2 market implementation comprises different steps that depend on each other and must be executed in a particular order. This section describes the timing of different events that take place when the market is operational.

Figure 4 in the workflow document [1] gives an overview of how the three modules of the real-time price engine are timed. However, those specifications have changed, cf. Section 3.1, and we will provide an overview of how the timing of different modules is has actually been implemented in the phase 2 market.

The timing for broadcasting the daily and hourly forecasts as well as for the real-time market price follows the definition of what is described in D1.7 [2]

3.1.1 DAILY TASKS

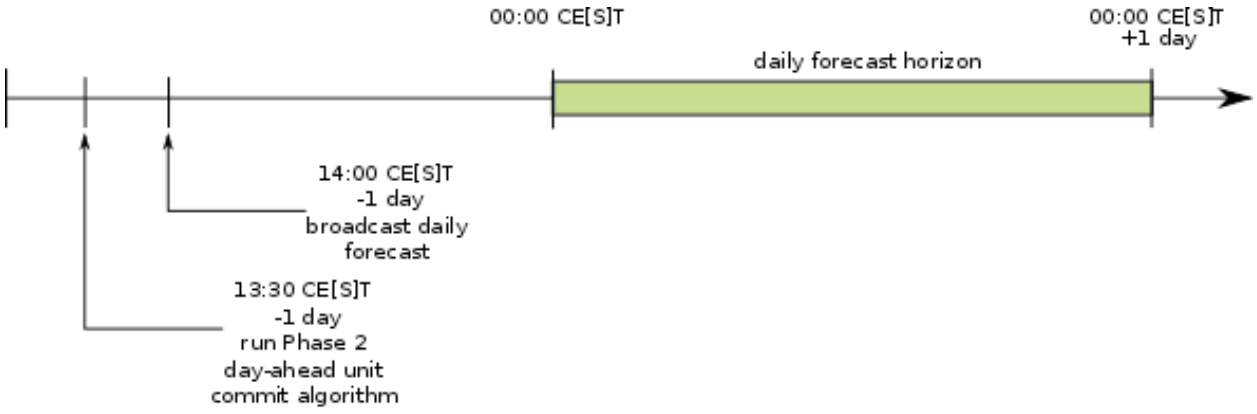

Figure 4 Timing of daily market events

Every day at 13.30h CE[S]T, the day-ahead unit commitment optimization (Step 1 of the real-time price engine) is executed to compute the day-ahead generator production levels and states. At 14:00h CE[S]T the day-ahead forecast is broadcasted to the participants.

3.1.2 HOURLY TASKS

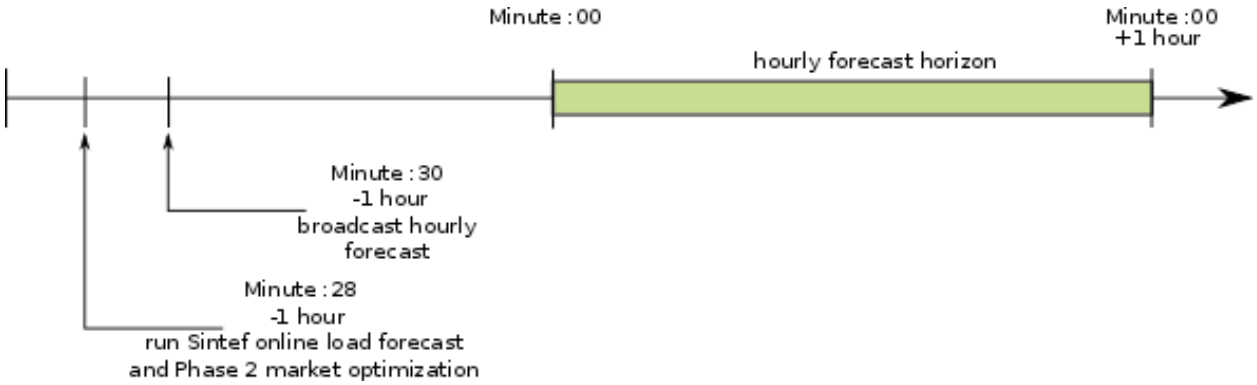

Figure 5 Timing of hourly market events

At minute 28 of every hour (e.g. 14.28h, 15.28h,…) the EcoGrid market optimization (Step 2) is executed to compute the hourly price forecast for the next 3 hours (i.e. 15.00-18.00h, 16.00-19.00h,…). Since step 2 relies on the load forecast for the same time interval produced by the load forecasting module, the latter has to be executed on an hourly basis before the the step 2 as well. Load forecasting is performed based on the current load model states that are updated every 5 minutes when a new aggregate load measurement and the commit price become available. At minute 30 of every hour the first 12 values out of the calculated 36 values are broadcasted to the participants as real time market price hourly forecast. If the calculation of the load forecast and the market optimization would take to long to finish before the broadcast deadline (minute 30) the market will broadcast the spot market price vaild for the hourly forecast horizon.

3.1.3 5 MINUTE TASKS

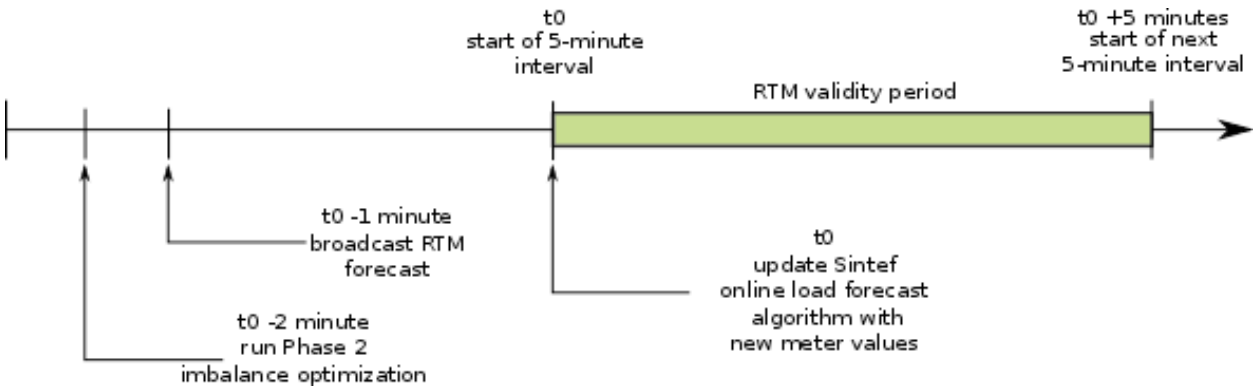

Figure 6 Timing of 5-minute market events

Finally, two minutes before every start of a 5-minute interval the imbalance optimization (Step 3 of the phase 2 real-time price engine) is executed based on the current balance value and the hourly real time market price forecast, which was calculated at minute 28 the hour before, valid for the upcomming 5-minute interval. The output of step 3 is the 5 minute real-time price that is commited at the same frequency. This calculated real-time market price is then broadcasted at one minute before the start of the interval. If the imbalance optimization calculation is not finished before the broadcast deadline the market will broadcast the spot price valid for the upcomming 5-minute interval.

3.1.4 MARKET DATA IMPORTER

To execute the above steps the market needs external inputs as the Nordpool Spot Price, wind data from EnFor and weather information from the Danish Metrological Institute (DMI). The inputs are described in detail in [1].

The market has an subsystem for data collection which runs an importer task for every data source. The different inputs are checked for new data at differend rates shown in below table. If new data is available it will be imported to corresponding table in the data warehouse.

Importer name	Type	Interval	Description
Weather Forecast	http	10 min	Weather forecast data from DMI
Spot Price	ftp	10 min	Spot Price for DK2 from Nordpool
EnFor Wind Data	ssh	1 min	Enfor wind prediction and wind simulation data

Table 10 Data sources and polling interval

3.2 TIMING OF STEP 2: 30MIN VS 60MIN

The original description given in the workflow document [1] describes the Phase 2 Market algorithm with two 30 minute forecasts which is contrary to what is described in D1.7 [2] which describes one 60 minute forecast broadcasted at minute 30 and valid from minute 0 to 60 of the next hour as also seen in 3.1.2. This time was also used during the Phase 1 Market and all test cases until now. During the implementation of the Phase 2 Market it was therefore decided to stick to the current timing described in D1.7.

The only change in the OPL code needed for this is in the Market Model (step) where some array indices need to be adopted the following way.

Change:

```
p_ur_initial <- p_ur[g][6]

p_dr_initial <- p_dr[g][6]
```

to

```
p_ur_initial <- p_ur[g][12]

p_dr_initial <- p_dr[g][12]
```

Emil Mahler Larsen kindly provided us the needed change.

3.3 VARIABLE STEP 2 FORECAST HORIZON LENGTH BASED ON METER MEASUREMENT AVAILABILITY

The Sintef online load forecast algorithm is comprised out of two parts. The update part runs every 5 minutes (see also 2.2.3) and the forecast method that runs before the market optimization (step 2) OPL as it provides the load forecast input to it. Figure 7 shows an overview of the timings. The load forecast module runs at minute 28 as a prerequisite of the market optimization OPL (step 2) which needs 36 intervals from minute 0 of its validity period on (green bar in figure). This leaves a gap, which is at least 35 minutes long as at best the online load forecast module was updated with meter data up to minute 25. In practice meter data is on average 8 to 10 minutes behind the current time therefore the pre-forecast will be 8 to 10 intervals long which means the overall forecasting period is 44 to 46 intervals or 220 to 230 minutes long. In case of a meter data outage the per-forecast period

increases up to 288 intervals, which is one day. Above 288 intervals forecast calculation is not started, as it would not finish within the broadcast deadline. The load forecast is a prerequisite to the market optimization OPL (step 2) it is also not started an the spot price valid for the upcoming hourly real-time market price forecast is used.

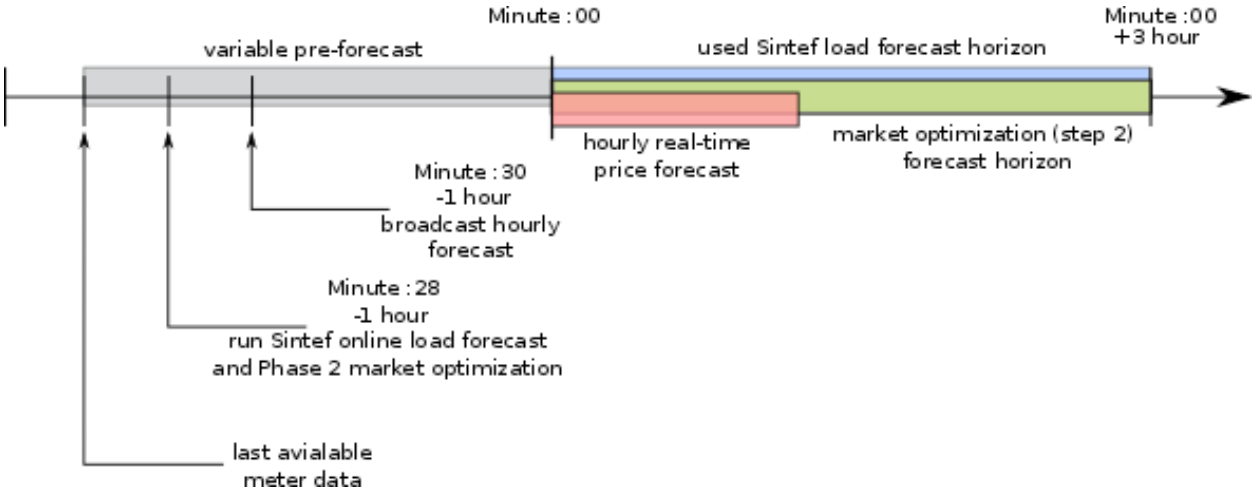

Figure 7 Variable step 2 forecast timing diagram

4 MARKET INITIALIZATION

As seen in 3 the Phase 2 EcoGrid Real-Time Market follows a given sequence of timed events to calculate the different necessary steps for generating a price. As the timeline for such events is very long with events even occurring once a day it needs a solution to start or restart the market for the first time or after an outage period. The document providing the specification [1] is not explaining this.

The initial startup of the EcoGrid Real-Time Market and the restarting of it follow the same procedure with one exception in the Sintef load forecast module, which needs to be initially trained. The startup sequence of the EcoGrid Real-Time Market is the following. After startup and property reading the broadcast timers are initialized first to be able to send prices and forecasts to next occurring market event. If the market startup is not yet complete at that time the spot price will be sent as a default for the rest of if the initialization period. Next the day-ahead unit commit algorithm (step 1) is executed for the current day if there is not yet a result for the current day in the corresponding database table. As the name day-ahead unit commit algorithm suggest this step is usually running the day before (see 3.1.1) and therefore the input data is taken from the database as it would be the day before at 13:30 CE[S]T. Next if it is after 13:30 CE[S]T and there a no results for the day-ahead commit algorithm yet in the corresponding database table for the next day also the day-ahead commit for the next day is calculated. This time with data as of 13:30 CE[S]T today.

In the next step the Sintef load forecast algorithm will be updated with meterreadings up to minute 25 of the last hour if needed or if it is the initial start of the EcoGrid Real-Time Market Phase 2 the Sintef load forecast module is trained with the data of the last 2016 (one week) intervals before minute 25 of the last hour. After updating the load forecast module the hourly tasks as described in 3.1.2 are executed for the current hour if there is not yet a result in the corresponding table in the database. If it is after minute 28 and no result for the next hour of the hourly task is in the corresponding table in the database the updating of the Sintef load forecast module and the calculation of the hourly tasks is repeated for the next hour.

As the second to last step the real-time market price is calculated for the next 5-minute interval. Finally the timer for the daily, hourly and 5-minute interval Phase 2 tasks are initialized and started.

Independent of the EcoGrid Real-Time Market Phase 2 module the datacollector modules are initialized and started.

5 MARKET PERSISTENCE

5.1 SINTEF LOAD FORECAST MODULE

For the Sintef load forecast module the following data is persisted to the database

5.1.1 MEASUREMENTS

The Sintef load forecast algorithm is update every 5 minutes with new latest available meter data as well as the price and temperature data for the meter interval. In addition some organizational fields are written to the database to a table called eg_mkt_phase2_measurements.

Column name	Type	Description
UUID	uuid	UUID of entry, used in States table to reference the entry
TS	timestamp	Timestamp of entry to the database
MEAS_TS	timestamp	Timestamp defining the start of the interval of this measurement
IDX_T	integer	Long Term Trend Data Index used see 2.1.1.1
Z_T	double	Power Measurement for this interval
OUTTEMP	double	Outside temperature according to DMI
PRICE	double[]	RTM Price for this interval and the interval before

5.1.2 STATES

The Sintef load forecast algorithm keeps state between the different update and forecast operations. To keep this state also over market restarts and for evaluation purposes the states are written to the database table eg_mkt_phase2_states.

Column name	Type	Description
UUID	uuid	UUID of entry
LOOP_UUID	uuid	UUID to identify the update run
TS	timestamp	Timestamp of entry to the database
EPS	double	Sintef algorithm variable
S1	double[]	Sintef algorithm variable
S2	double[]	Sintef algorithm variable
U	double[]	Sintef algorithm variable
V	double[]	Sintef algorithm variable
W	double[]	Sintef algorithm variable
X	double[]	Sintef algorithm variable
MEAS_TS	timestamp	last measurement interval time used in last update or forecast step
MEAS_UUID	uuid	UUID of last measurement

STATE_UUID	uuid	UUID identifying the history

5.1.3 FORECAST

The Sintef load forecast which is calculated prior to the market optimization (hourly task , see 3.1.2) is also persisted for evaluation purposes into eg_mkt_phase2_forecast

Column name	Type	Description
UUID	uuid	UUID of entry
TS	timestamp	Timestamp of entry to the database
EPS	double[]	Sintef algorithm variable
Y	double[]	Sintef algorithm variable
Z	double[]	Sintef algorithm variable
S1	double[]	Sintef algorithm variable
S2	double[]	Sintef algorithm variable
U	double[]	Sintef algorithm variable
V	double[]	Sintef algorithm variable
W	double[]	Sintef algorithm variable
PRICE	double[]	Price forecast from last hourly task, alternatively spot price
OUTTEMP	double[]	Outside temperature forecast from DMI
LOOP_UUID	uuid	UUID of run
STATE_UUID	uuid	UUID identifying the state history

5.1.4 SINTEF PARAMETER

The load forecast algorithm uses a number of static parameters for the update and forecast step. This parameters are persisted on change to the database table eg_mkt_phase2_parameter manually and configured in the market module configuration file by using the identifying UUID.

Column name	Type	Description
UUID	uuid	UUID of entry
TS	timestamp	Timestamp of entry to the database
NUM_SLOT_DAY	integer	Sintef algorithm parameter
NUM_SLOT_WEEK	integer	Sintef algorithm parameter
SPERIOD	integer	Sintef algorithm parameter
IDX_QS	integer	Sintef algorithm parameter
IDX_PS	integer	Sintef algorithm parameter
IDX_P	integer	Sintef algorithm parameter
IDX_Q	integer	Sintef algorithm parameter
IDX_T_INIT	integer	Sintef algorithm parameter
GAMMA1	double	Sintef algorithm parameter
GAMMA2	double	Sintef algorithm parameter
RESP	double[]	Sintef algorithm parameter
RESQ	double[]	Sintef algorithm parameter

RESPS	double[]	Sintef algorithm parameter
RESQS	double[]	Sintef algorithm parameter
LTTREND	double[]	Sintef algorithm parameter
BETAS	double[]	Sintef algorithm parameter
S1_INIT	double[]	Sintef algorithm parameter
S2_INIT	double[]	Sintef algorithm parameter
U_INIT	double[]	Sintef algorithm parameter
V_INIT	double[]	Sintef algorithm parameter
W_INIT	double[]	Sintef algorithm parameter

5.2 DAY-AHEAD UNIT COMMIT (STEP 1 OPL)

For the day-ahead unit commit algorithm, which is the daily step in the real time market price calculation, all inputs and outputs to the optimizations are logged into the database.

5.2.1 DAY-AHEAD UNIT COMMIT INPUT DATA

The input data to the optimization of the day-ahead unit commit is persisted into the eg_mkt_phase2_opt_step1_in table in the database. For a description of the OPL input variables refer to [1]

Column name	Type	Description
UUID	uuid	UUID of entry
TS	timestamp	Timestamp of entry to the database
LOOP_UUID	uuid	UUID to identify the run and the output as well as the solving state information
OPERATIONAL_DAY	timestamp	Start timestamp of the operational day of this step
NTIME	integer	OPL input variable
NGEN	integer	OPL input variable
NFIVE	integer	OPL input variable
SCALING_FACTOR	double	OPL input variable
GENERATOR_TOTAL_CAPACITY	double	OPL input variable
COST_LOAD_SHED	double	OPL input variable
COST_WIND_SPILL	double	OPL input variable
GAP	double	OPL input variable
RAMP	double	OPL input variable
WEEK_LOAD	double[]	OPL input variable
P_DA_WIND	double[]	OPL input variable
PARA_UUID	uuid	UUID of used parameter set for step1

5.2.2 DAY-AHEAD UNIT COMMIT OUTPUT DATA

The output of the day-ahead unit commit optimization is persisted into the eg_mkt_phase2_opt_step1_out table in the database. For a description of the OPL output variables refer to [1]

Column name	Type	Description
UUID	uuid	UUID of entry

TS	timestamp	Timestamp of entry to the database
OPERATIONAL_DAY	timestamp	Start timestamp of the operational day of this step
LOOP_UUID	uuid	UUID to identify the run and the input as well as the solving state information
P_SCH	double[]	OPL output variable
P_DA_WIND	double[]	OPL output variable
Q_MAX	double[]	OPL output variable
PA_DA_LOAD_RAW	double[]	OPL output variable
P_DA_LOAD	double[]	OPL output variable
NGEN	integer	OPL output variable
SCALING_FACTOR	double	OPL output variable

5.2.3 DAY-AHEAD UNIT COMMIT PARAMETER

The parameters of the day-ahead unit commit are stored in eg_mkt_phase2_opl_step1_para and configured in the market property file by the identifying uuid of database entry.

Column name	Type	Description
UUID	uuid	UUID of entry
NTIME	integer	OPL parameter variable
NGEN	integer	OPL parameter variable
NFIVE	integer	OPL parameter variable
SCALING_FACTOR	double	OPL parameter variable
GENERATOR_TOTAL_CAPACITY	double	OPL parameter variable
COST_LOAD_SHED	double	OPL parameter variable
COST_WIND_SPILL	double	OPL parameter variable
GAP	double	OPL parameter variable
RAMP	double	OPL parameter variable

5.3 MARKET OPTIMIZATION (STEP 2)

For the market optimization algorithm, which is the hourly step in the real time market price calculation, all inputs and outputs to the optimizations are logged into the database.

5.3.1 MARKET OPTIMIZATION INPUT

The input data to the optimization of the market optimization is persisted into the eg_mkt_phase2_opt_step2_in table in the database. For a description of the OPL input variables refer to [1]

Column name	Type	Description
UUID	uuid	UUID of entry
TS	timestamp	Timestamp of entry to the database
OPERATIONAL_DAY	timestamp	Start timestamp of the operational day of this step
LOOP_UUID	uuid	UUID to identify the run and the output as well as the solving state information

STEP1_INPUT	uuid	UUID of step1 which holds the step1 input data used for this step2
STEP1_INPUT2	uuid	As above but for UUID for the next day if the step horizons spans 2 days
STEP1_OUTPUT	uuid	UUID of step1 which holds the step1 output data used for this step2
STEP1_OUTPUT2	uuid	As above but for UUID for the next day if the step horizons spans 2 days
FORECAST_UUID	uuid	UUID identifying the Sintef load forecast results used for this step
NTIME	integer	OPL input variable
NLOAD	integer	OPL input variable
MIN_ON	integer	OPL input variable
COST_WIND_SPILL	double	OPL input variable
COST_LOAD_SHED	double	OPL input variable
COST_FREQUENCY_CONTROL	double	OPL input variable
FLEX_LOAD	double	OPL input variable
BID_PERCENT	double	OPL input variable
RAMP_RATE	double	OPL input variable
NGEN	integer	OPL input variable
SCALING_FACTOR	double	OPL input variable
P_SCH	double[]	OPL input variable
Q_MAX	double[]	OPL input variable
SPOT_PRICE	double[]	OPL input variable
P_REF_RAW	double[]	OPL input variable
ALPHA	double[]	OPL input variable
CHANGEL_MIN_RAW	double[]	OPL input variable
CHANGEL_MAX_RAW	double[]	OPL input variable
P_DA_WIND	double[]	OPL input variable
BALANCE	double[]	OPL input variable
P_UR_INITIAL	double[]	OPL input variable
P_DR_INITIAL	double[]	OPL input variable

5.3.2 MARKET OPTIMIZATION OUTPUT

The output of the market optimization is persisted into the eg_mkt_phase2_opt_step2_out table in the database. For a description of the OPL output variables refer to [1]

Column name	Type	Description
UUID	uuid	UUID of entry
TS	timestamp	Timestamp of entry to the database
OPERATIONAL_DAY	timestamp	Start timestamp of the operational day of this step
LOOP_UUID	uuid	UUID to identify the run and the output as well as the solving state information
PRICE_RT_FORECAST	double[]	OPL output variable
P_UR_INITIAL	double[]	OPL output variable

P_DR_INITIAL	double[]	OPL output variable
NLOAD	integer	OPL output variable
P_UR	double[]	OPL output variable
P_DR	double[]	OPL output variable
QUANT_UP	double[]	OPL output variable
QUANT_DO	double[]	OPL output variable
CHANGEL	double[]	OPL output variable
X	integer	OPL output variable
Z	integer	OPL output variable

5.3.3 MARKET OPTIMIZATION PARAMETER

The parameters of the market optimization are stored in eg_mkt_phase2_opl_step2_para and configured in the market property file by the identifying uuid of database entry.

Column name	Type	Description
UUID	uuid	UUID of entry
NTIME	integer	OPL parameter variable
NLOAD	integer	OPL parameter variable
MIN_ON	integer	OPL parameter variable
COST_WIND_SPILL	double	OPL parameter variable
COST_LOAD_SHED	double	OPL parameter variable
COST_FREQUENCY_CONTROL	double	OPL parameter variable
FLEX_LOAD	double[]	OPL parameter variable
BID_PERCENT	double	OPL parameter variable
RAMP_RATE	double	OPL parameter variable
NGEN	double	OPL parameter variable
SCALING_FACTOR	double	OPL parameter variable
CHANGEL_MAX_RAW	double[]	OPL parameter variable
CHANGEL_MIN_RAW	double[]	OPL parameter variable

5.4 IMBALANCE OPTIMIZATION (STEP 3)

For the imbalance optimization algorithm, which is the 5-minute step in the real time market price calculation, all inputs and outputs to the optimizations are logged into the database.

5.4.1 IMBALANCE OPTIMIZATION INPUT

The input data to the imbalance optimization is persisted into the eg_mkt_phase2_opt_step3_in table in the database. For a description of the OPL input variables refer to [1]

Column name	Type	Description
UUID	uuid	UUID of entry

TS	timestamp	Timestamp of entry to the database
OPERATIONAL_TIME	timestamp	Start timestamp of the operational interval of this step
LOOP_UUID	uuid	UUID to identify the run and the output as well as the solving state information
INPUT_STEP2	uuid	UUID of step2 input which holds the step2 input data used for this step
OUTPUT_STEP2	uuid	UUID of step2 which holds the step2 output data used for this step3
NLOAD	integer	OPL input variable
SCALING_FACTOR	double	OPL input variable
CHANGEL	double[]	OPL input variable
PRICE_RT_FORECAST	double[]	OPL input variable
BALANCE	double	OPL input variable
ALPHA	double[]	OPL input variable
CHANGEL_MAX_RAW	double[]	OPL input variable
CHANGEL_MIN_RAW	double[]	OPL input variable
SPOT_PRICE	double	OPL input variable

5.4.2 IMBALANCE OPTIMIZATION OUTPUT

The output of the imbalance optimization is persisted into the eg_mkt_phase2_opt_step3_out table in the database. For a description of the OPL output variables refer to [1]

Column name	Type	Description
UUID	uuid	UUID of entry
TS	timestamp	Timestamp of entry to the database
OPERATIONAL_TIME	timestamp	Start timestamp of the operational interval of this step
LOOP_UUID	uuid	UUID to identify the run and the input as well as the solving state information
PRICE_RT	double[]	OPL output variable
ALPHA	double[]	OPL output variable
CHANGEL	double[]	OPL output variable

5.4.3 IMBALANCE OPTIMIZATION PARAMETER

The parameters of the imbalance optimization are stored in eg_mkt_phase2_opl_step3_para and configured in the market property file by the identifying uuid of database entry.

Column name	Type	Description
UUID	uuid	UUID of entry
NLOAD	timestamp	OPL parameter variable
SCALING_FACTOR	double	OPL parameter variable
CHANGEL_MAX_RAW	double[]	OPL parameter variable

CHANGEL_MIN_RAW	double[]	OPL parameter variable

6 REFERENCES

[1] Rasmussen, C. B. *EcoGrid EU Demonstration phase 2 and 3 specification. Task force working document.* EcoGrid EU, 2013.

[2] Kok, K. *Deliverable 1.7 Business models, requirements and architecture specification.* EcoGrid EU 2012