ESSIM

Arun Subramanian, Selma Causevic, Edwin Matthijssen

Mar 29, 2024

CONTENTS:

Note: This documentation is work in progress and far from finished. New sections will be added in the near future. Whenever questions or feedback is received from end users, we're trying to update this documentation on the fly. So don't hesitate to contact us, whenever you run into problems.

CHAPTER

ONE

INTRODUCTION

Note: This documentation is work in progress and far from finished. New sections will be added in the near future. Whenever questions or feedback is received from end users, we're trying to update this documentation on the fly. So don't hesitate to contact us, whenever you run into problems.

The Energy System Simulator (ESSIM) is a discrete time simulation tool and collection of models that calculates energy flows in assets and the effects thereof, in an interconnected hybrid energy system over a period of time. With the help of the energy flows ESSIM calculates, one can get insights into how well the assets in a network are dimensioned, if there is overloading in any given transport asset (like pipe, cables, etc.) and what the effect of storage is in any part of the network.

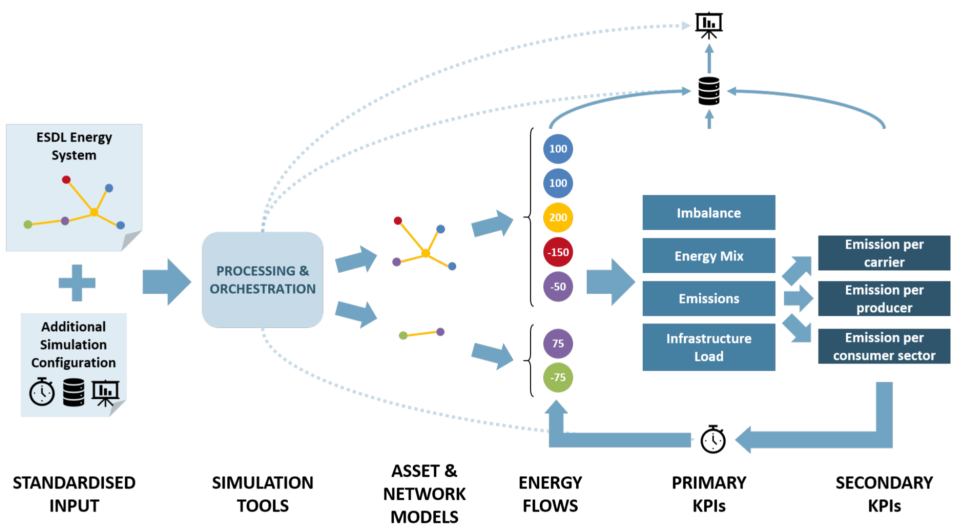


Fig. 1: An overview of what ESSIM does

Traditionally, dimensioning of assets was assessed by looking at yearly values of energy supply and demand. In a simple network where a producer supplied 10TJ per year and a consumer consumed 10TJ, the system, under such a calculation, is in balance and the energy producer is properly dimensioned to meet the demands of the consumers in the network.

However, in reality producers do not produce a constant amount of energy every hour of every day throughout the year. Neither do demands have a constant flat line consumption pattern. These variations when captured as a profile highlight the mismatch despite the annual sums adding up.

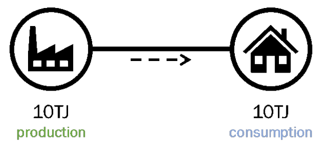


Fig. 2: A system that appears to have no net yearly energy imbalance

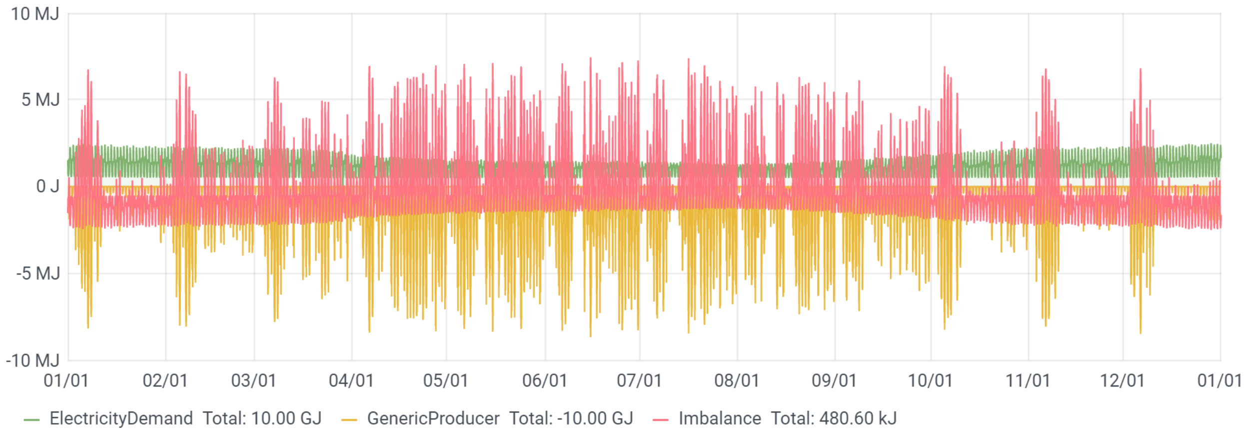


Fig. 3: Hourly data showing imbalance

ESSIM helps the user identify such nuances as it repeatedly calculates energy flows per time step for a simulation period. It takes as inputs the energy system defined in ESDL and calculates optimal schedule of flexible producers and the effect of this schedule in terms of emissions, costs, load on the network, etc. Thanks to the easy configurability of the energy system with the help of the ESDL MapEditor, the user can use ESSIM to perform "what if?" scenario analyses on current and future energy systems. Along with the primary KPIs (Key Performance Indicator) of ESSIM (Energy mix, network imbalance, emissions, etc.), external KPI modules connected to ESSIM also allow for post-processing ESSIM data to get measurable insights into the energy system variations.

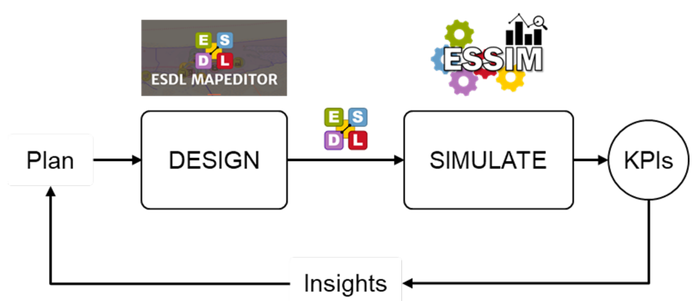


Fig. 4: ESSIM used in the planning-design phase of Energy Transition

At the heart of the tool are:

- A simulation engine that orchestrates a fixed discrete timestep evaluation of each commodity network's balance in a pre-determined order,
- A flexibility-based demand-supply matching algorithm that uses marginal costs of energy production as a means to grade desirability of producers,

- A tree-based transport network solver that calculates the load on various transport elements based on the demand-supply solution determined above.

The tool outputs the data provided and generated during the course of the simulation into a time-series database (InfluxDB) and generates a dashboard (Grafana) to visualise the same. ESSIM allows for connecting external asset models (via an MQTT interface) and external network (solver) models (via a REST interface), alongside its internal models and while using its own orchestration mechanism. This makes ESSIM a viable candidate for a co-simulation orchestrator.

CHAPTER

TWO

BASIC PRINCIPLES

Note: This documentation is work in progress and far from finished. New sections will be added in the near future. Whenever questions or feedback is received from end users, we're trying to update this documentation on the fly. So don't hesitate to contact us, whenever you run into problems.

2.1 Energy Assets

ESSIM follows the ESDL principle of classifying energy system assets into Producers, Consumers, Storages, Conversions and Transports. Each asset (except for Transport assets) can either have a pre-determined behaviour (*inflexible*) programmed into it with the help of a profile, or let ESSIM calculate its energy allocation (*flexible*) based on its flexibility, some control strategy and the behaviour of other assets in its network. ESSIM contains models for each of these five classes of assets and thanks to ESDL's hierarchical data structure, model behaviour can also be inherited from parent models. E.g. With a generic Consumer behaviour implementation, all existing ESDL Consumers such as ElectricityDemand, HeatingDemand, etc. and any ESDL Consumers added in the future can be supported within ESSIM. Additionally, implemented within ESSIM are also some specific asset models that have certain peculiarities. E.g. A co-generation plant is a conversion asset with two output ports with one output effecting the other.

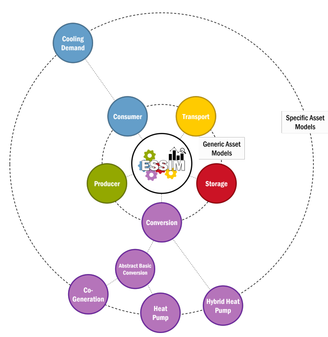


Fig. 1: All implemented asset models in ESSIM

2.2 Transport Network

In ESSIM, a transport network is an atomic network of connected energy assets perusing the same energy carrier. This means ESSIM treats isolated assets physically connected only to each other but using the same energy carrier as separate transport networks. Internally ESSIM creates a tree of such a network where the Producer/Consumer/Conversion/Storage assets are the leaf nodes and the Transport assets are the branches. Conversion assets appear in multiple transport networks as consumers of some energy carriers and producers of some others. Based on the provided ESDL description and configured behaviour, each transport network attempts to balance itself in each time step.

The transport network trees are created in the pre-processing step in ESSIM and cannot be changed during the simulation. In addition to creating the trees, ESSIM also pre-determines the order and parallelisation in computing the network tree balances.

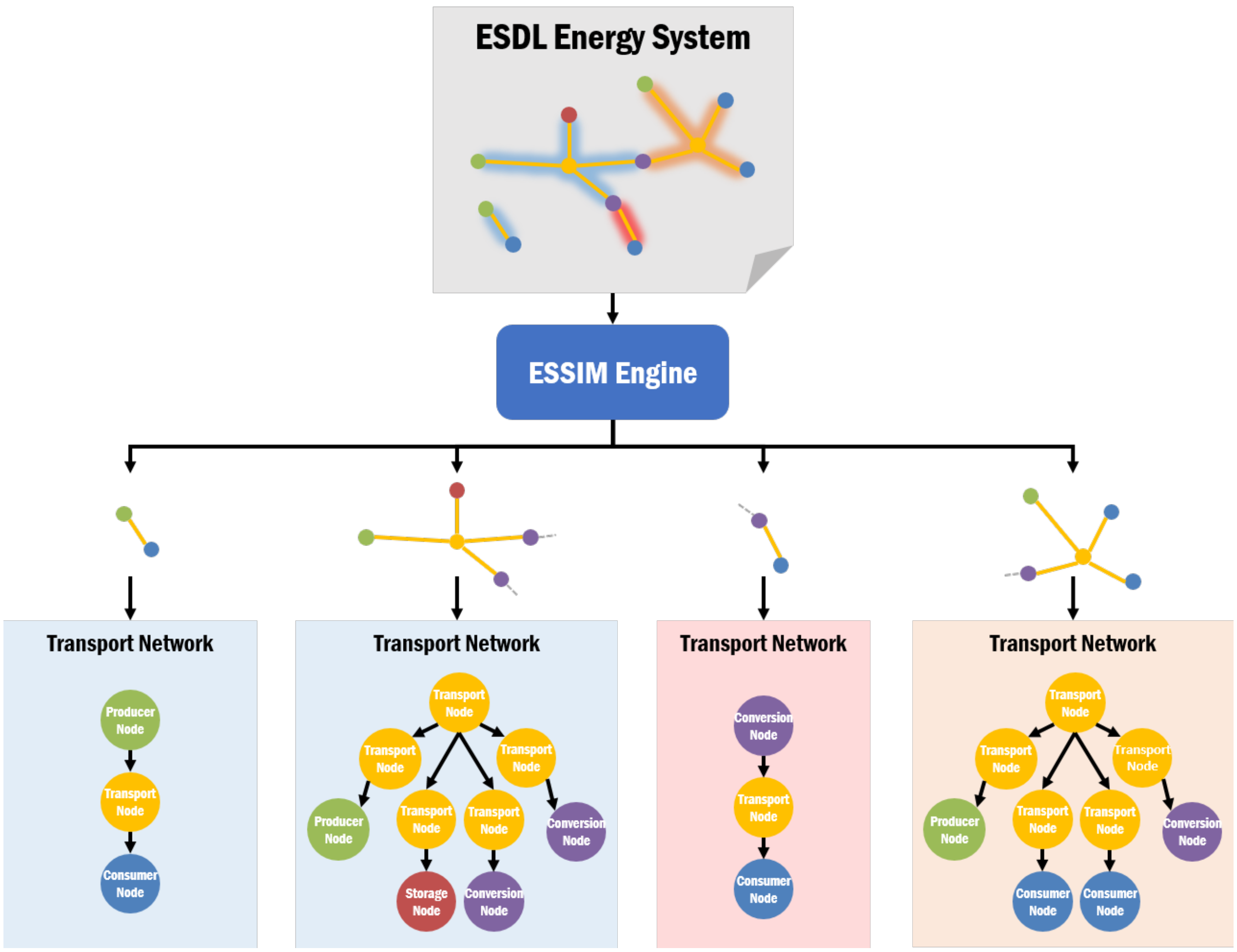


Fig. 2: Splitting of an energy system into constituent transport networks

CHAPTER THREE

CONFIGURATION

Note: This documentation is work in progress and far from finished. New sections will be added in the near future. Whenever questions or feedback is received from end users, we're trying to update this documentation on the fly. So don't hesitate to contact us, whenever you run into problems.

Below are a list of ESDL attributes that have a direct effect on the way an asset is simulated in ESSIM. As explained before, ESDL assets are hierarchical in nature and hence the attributes defined for an asset class here can also be inherited by any of its sub-classes. For eg, configuring an ElectricityDemand or HeatingDemand is identical to configuring a Consumer (with the only difference being the EnergyCarrier it consumes).

Table 1: ESDL Attributes and their effect in ESSIM

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
Consumer	Flexible	Power	Rated input power in Watts	0	Demand will be scheduled by ESSIM between 0% and 100% of this power.
		Marginal Costs	Cost to reduce consumption by 1W normalised to the range [0,1]	1	It determines a priority in fulfilment of demand among consumers; A consumer with higher marginal costs has a higher priority over others.

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
	Inflexible	Profile	Demand Profile	N/A	One of DateTimeProfile, SingleValueProfile or InfluxDBProfile containing a time-varying demand pattern of power or energy attached to the Input Port of the consumer.
	Both	Commissioning Date	Date (and time) of commissioning of asset	Disabled	For the simulation period before this date, the asset would be "disabled" and gets a "zero" allocation.
		Decommissioning Date	Date (and time) of decommissioning of asset	Disabled	For the simulation period after this date, the asset would be "disabled" and gets a "zero" allocation.
		Name	Human-readable name for the asset	<ESDLClass>_<first-4-chars-of-id>	This property is used to annotate the asset in the Grafana dashboard. If not available, it will fall back to asset's ID.
		Sector	Sector the consumer asset belongs to	Default	This is to annotate a consumer to a sector it belongs to (such as Residential, Healthcare, Commercial, etc.). ESSIM uses this information to divide the total system emissions into these sectors.

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
Conversion	Flexible	Power	Rated output power in Watts	0	Conversion asset will be scheduled by ESSIM between 0% and 100% of this power.
		Marginal Costs	Cost to raise production or reduce consumption by 1W normalised to the range [0,1]	0.5	Determines a priority in fulfilment of demand among consumers (on the Input side of Conversion) or a priority in scheduling of producers (on the Output side of Conversion). A producer with lower marginal costs and a consumer with higher marginal costs have a higher priority over others.
		Efficiency	Conversion efficiency of asset (See also Behaviour)	0.6 (60%)	Rated efficiency of the conversion asset in converting input energy carrier to output energy carrier.

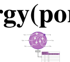
continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
		Control Strategy	Strategy to operate this asset	N/A	One of either "DrivenByDemand" or "DrivenBySupply" designated with the appropriate port. If "DrivenByDemand" is chosen, then an Output Port must be attached to it. Then this conversion asset will attempt to fulfil the demand placed for the energy carrier at this output port by being a flexible producer. The MarginalCosts determine the priority of this asset while competing with other producers in the network. If "DrivenBySupply" is chosen, then an Input Port must be attached to it. Then this conversion asset will attempt to consume the supply of energy carrier placed at this input port by being a flexible consumer. The MarginalCosts determine the priority of this asset while competing with other consumers in the network.

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
	Inflexible	Control Strategy	Strategy to operate this asset	N/A	Set to “DrivenByProfile” and attach one of DateTimeProfile, SingleValueProfile or InfluxDBProfile containing a time-varying demand/production pattern of power or energy to it.
	Both	Behaviour	Behaviour of this asset as ESDL InputOutputRelation. **Applicable only for Conversion assets with multiple input and output ports. For single input-output conversions, use the Efficiency parameter**	N/A	This is a way to model the (linear) relationship between energy produced/consumed at the ports of a multi-input multi-output conversion asset with respect to one main port. Fill in a positive ratio to describe the port ratio such that **energy(mainPort) = ratio * energy(port)** 
		Commissioning Date	Date (and time) of commissioning of asset	Disabled	For the simulation period before this date, the asset would be “disabled” and gets a “zero” allocation.
		Decommissioning Date	Date (and time) of decommissioning of asset	Disabled	For the simulation period after this date, the asset would be “disabled” and gets a “zero” allocation.

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
		Name	Human-readable name for the asset	<ESDLClass>_<first-4-chars-of-id>	This property is used to annotate the asset in the Grafana dashboard. If not available, it will fall back to asset's ID.
Heat Pump *(Heat Pump is a Conversion asset. So all attributes set for a Conversion asset apply to it as well. Only specific properties will be mentioned in this table)*	Both (Flexible and Inflexible)	COP	Coefficient of performance of the heat pump	3.5	This is used in computing the energy input/output at the various ports of the heat pump like so: COP = HeatOut/ElecIn HeatOut = ElecIn + HeatIn **Note: Efficiency parameter of a heat pump is ignored**
		Carriers	Energy carriers supported by this asset	N/A	Heat pump currently supports only these commodities: 
Co-generation *(Co-generation is a Conversion asset. So all attributes set for a Conversion asset apply to it as well. Only specific properties will be mentioned in this table)* *PS: Only a Heat-Electricity co-generation plant is currently supported*	Both (Flexible and Inflexible)	Heat Efficiency	Efficiency of heat production process	0.35	This is used in computing the heat generated or the fuel consumed to generate x Joules of heat
		Electrical Efficiency	Efficiency of electricity production process	0.55	This is used in computing the electricity generated or the fuel consumed to generate x Joules of electricity

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
		Carriers	Energy carriers supported by this asset	N/A	Co-generation currently supports only these output commodities: 
Producer	Flexible	Power	Rated input power in Watts	0	Production will be scheduled by ESSIM between 0% and 100% of this power.
		Marginal Costs	Cost to raise production by 1W normalised to the range [0,1]	0.5	It determines a priority in fulfilment of demand among consumers; A producer with lower marginal costs has a higher priority over others.
		Control Strategy	Strategy to operate this asset	N/A	A producer may be designated with a CurtailmentStrategy with a MaxPower attribute. Then, the producer output is limited to MaxPower.
	Inflexible	Profile	Production Profile	N/A	One of DateTimeProfile, SingleValueProfile or InfluxDBProfile containing a time-varying demand pattern of power or energy attached to the Output Port of the producer.

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
	Both	Commissioning Date	Date (and time) of commissioning of asset	Disabled	For the simulation period before this date, the asset would be “disabled” and gets a “zero” allocation.
		Decommissioning Date	Date (and time) of decommissioning of asset	Disabled	For the simulation period after this date, the asset would be “disabled” and gets a “zero” allocation.
		Name	Human-readable name for the asset	<ESDLClass>_<first-4-chars-of-id>	This property is used to annotate the asset in the Grafana dashboard. If not available, it will fall back to asset’s ID.
Storage	Flexible	Fill Level	Initial state of charge of the storage	0	Initial fill level represented as State of Charge of the storage asset.
		Capacity	Capacity of the storage in Joules	0	Determines when the storage asset is full and cannot charge any more.
		Max Charge Rate	Maximum charge rate of the storage in Joules/second (Watts).	0	Storage asset is flexible to charge anywhere between 0% and 100% of this rate capped at remaining storable capacity.

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
		Max Discharge Rate	Maximum discharge rate of the storage in Joules/second (Watts).	0	Storage asset is flexible to discharge anywhere between 0% and 100% of this rate capped at remaining dischargable capacity.

continues on next page

Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
		Control Strategy	Strategy to operate this asset	N/A	A storage asset may be designated with a StorageStrategy with two marginal costs defined – marginal charging costs and marginal discharging costs (Cost to increase discharge or reduce charging by 1W normalised to the range [0,1]). Marginal charging costs determine the priority of this storage asset while competing with other consumers in the network. Marginal discharging costs determine the priority of this storage asset while competing with other producers in the network. **Attention: Marginal charging cost should always be lesser than marginal discharging cost!**

continues on next page

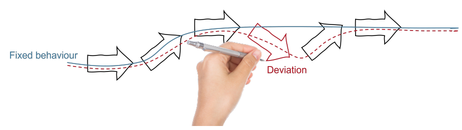
Table 1 – continued from previous page

ESDL asset	Mode of Operation	ESDL Attribute	Definition	Default	Effect in ESSIM
	Inflexible	Profile	Charge/Discharge profile	N/A	One of DateTimeProfile, SingleValueProfile or InfluxDBProfile containing a time-varying charge/discharge pattern of power, energy or state-of-charge attached directly to the storage. The same profile is to be used to define both charging and discharging behaviour.
	Both	Commissioning Date	Date (and time) of commissioning of asset	Disabled	For the simulation period before this date, the asset would be “disabled” and gets a “zero” allocation.
		Decommissioning Date	Date (and time) of decommissioning of asset	Disabled	For the simulation period after this date, the asset would be “disabled” and gets a “zero” allocation.
		Name	Human-readable name for the asset	<ESDLClass>_<first-4-chars-of-id>	This property is used to annotate the asset in the Grafana dashboard. If not available, it will fall back to asset’s ID.

CHAPTER FOUR

MODELLING FLEXIBILITY

Note: This documentation is work in progress and far from finished. New sections will be added in the near future. Whenever questions or feedback is received from end users, we're trying to update this documentation on the fly. So don't hesitate to contact us, whenever you run into problems.



Energy flexibility is the ability of an energy producing/consuming device to deviate from a planned or expected state. A device may be flexible in the amount of energy or time.

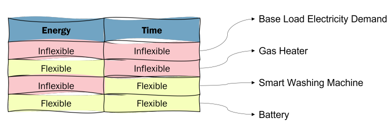


Fig. 1: Examples of flexible devices

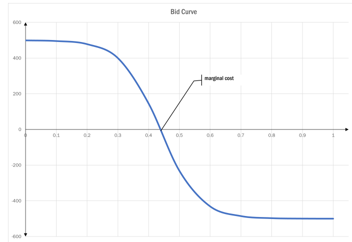
4.1 Bid Curves

ESSIM uses the language of bid curves to represent the flexibility of an asset. A bid curve is a monotonically decreasing, continuous function of energy over price. This *price* is different from a tariff in that it represents the need of a producer/consumer instead of the money paid or received. In ESSIM, this is expressed as a range of values between 0 and 1. An important feature of a bid curve is a marginal cost. This is the price beyond which consumption becomes unprofitable and/or production becomes profitable. To distinguish between production and consumption on a bid curve, energy production is represented as negative values and energy consumption as positive values. A flat bid curve represents inflexible behaviour.

An example bid curve of a prosumer (a device that can consume and produce energy, e.g. battery) with a marginal cost between 0.4 and 0.5.

Such a characterisation of energy flexibility allows for:

- Modelling a *rational* behaviour of devices in an energy system where producers try to maximise their profit and consumers try to minimise their expenses
- Preparing a merit-order of producers where cheaper producers are scheduled first, and one of consumers where more expensive consumers get a priority



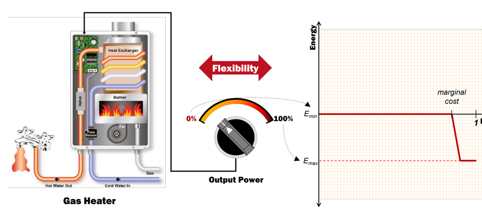
In ESSIM, this flexibility is used by the transport network model to balance demand and supply of energy in each time step.

The network model using bid curves to match demand and supply

Fact: In addition to allocation at the leaf nodes, the allocations at the branch (transport asset) nodes describe the energy flowing through that transport asset

4.1.1 Local ESSIM assets & bid curves

To construct a bid curve, two essential pieces of information about the device is needed - its capacity and its desirability at that time step. An asset's capacity to produce or consume energy is determined from its ESDL description, in case of flexible behaviour and from an attached profile, in case of inflexible behaviour. The exception to this rule is a storage asset whose flexibility at a given time depends on its state of charge at that moment. On the other hand, the desirability of an asset, designated by its marginal cost, is subjective to what the energy system modeller wants to achieve. For e.g, a highly polluting producer can be designated with a high marginal cost whereas renewable energy sources may have a marginal cost of 0.0. A behaviour of exporting only excess generation from a grid can be modelled by using a consumer with very low marginal cost. Such a consumer's demand will only be satisfied after the other highly paying consumers.

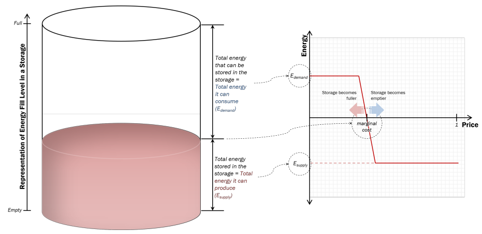


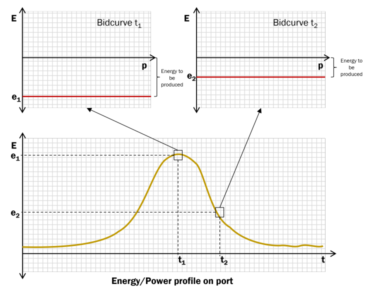
Creating the bid curve for a flexible gas heater. A curve with only a negative part indicates that this device is incapable of consuming energy. A high marginal cost indicates that it will be scheduled only after cheaper sources of heat such as heat pumps if present in the network.

Creating the bid curve for a storage. This curve has a positive and negative side determined by its state of charge and its maximum charging and discharging powers. The marginal cost in this case can be fixed or can keep changing based on the fill level of the storage.

In case the behaviour of a device is pre-determined to follow a certain profile, then the device is said to have no flexibility. In ESSIM, these devices produce flat bid curves. The amplitude of these curves change with respect to the value of the profile at that time step.

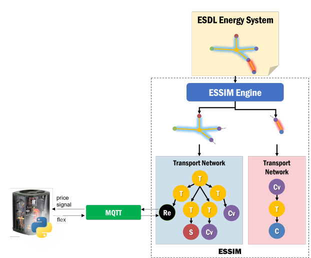
Bid curve for an inflexible device.





4.1.2 Remote assets & bid curves

ESSIM also has a provision to connect to a remote device model via MQTT. The skeleton for such a model, implemented in Python, can be found in this repository. With the help of such a model, more complex asset behaviours can be modelled compared to ESSIM's rudimentary model library as long as the model can create a bid curve for each time step representing its flexibility at that moment.



A remote (external) device model interacting with ESSIM.

CHAPTER

FIVE

ESSIM API

Note: This documentation is work in progress and far from finished. New sections will be added in the near future. Whenever questions or feedback is received from end users, we're trying to update this documentation on the fly. So don't hesitate to contact us, whenever you run into problems.

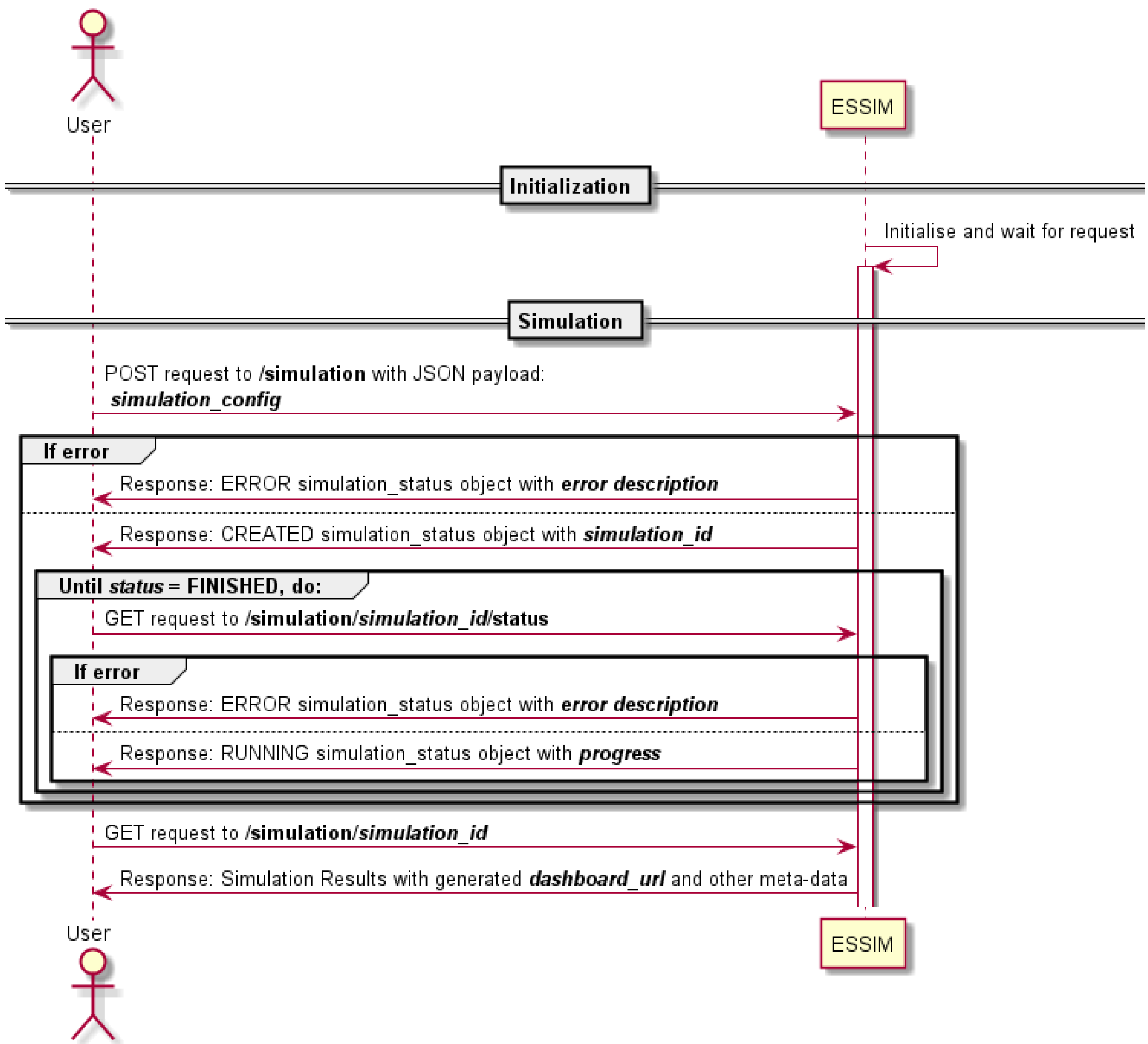
5.1 Sequence

ESSIM provides a REST API that allows users to interact with ESSIM for starting a simulation, requesting its progress, etc. The usual sequence to invoking a simulation and interacting with it is as follows:

5.2 APIs:

5.2.1 /simulation

- **HTTP Method**: POST
- **Description**: Create a new simulation
- **Request Body:**
    - *endDate*: End date of simulation in ISO-8601 format (`YYYY-MM-DDTHH:mm:ss±hh:mm`)
    - *esdlContents*: 64-bit encoded ESDL string
    - *influxURL*: URL of InfluxDB instance to store simulation results in
    - *scenarioID*: String ID representing the scenario being simulated. This ID is used to name the database in InfluxDB.
    - *simulationDescription*: Human-readable description of the simulation visible in the dashboard
    - *startDate*: Start date of simulation in ISO-8601 format (`YYYY-MM-DDTHH:mm:ss±hh:mm`)
    - *user*: Name of the user running the simulation. Used to tag the name of the Grafana dashboard
    - *csvFilesLocation*: (Optional) To export ESSIM simulation data into CSV, specify a location here
    - *mqttURL*: (Optional) To publish ESSIM data to an MQTT bus, specify the URL to an MQTT server here


User
ESSIM
Initialization
Initialise and wait for request
Simulation
POST request to /simulation with JSON payload:
simulation_config
If error
Response: ERROR simulation_status object with error description
Response: CREATED simulation_status object with simulation_id
Until status = FINISHED, do:
GET request to /simulation/simulation_id/status
If error
Response: ERROR simulation_status object with error description
Response: RUNNING simulation_status object with progress
GET request to /simulation/simulation_id
Response: Simulation Results with generated dashboard_url and other meta-data
User
ESSIM

- *amqpURL*: (Optional) To publish ESSIM data to an AMQP bus, specify the URL to an AMQP server here
- *kafkaURL*: (Optional) To publish ESSIM data to an Apache Kafka server, specify the URL to the server here
- *nodeConfig*: (Optional) To use a remote node with ESSIM, specify the following:
  - *esdlNodeId*: The ESDL ID of the asset represented by this node
  - *config*: Any key-value configuration to provide this node
  - *mqttHost*: MQTT Host URL
  - *mqttPort*: MQTT Port number
  - *mqttTopic*: Topic to reach the ESDL Node

Example:

```
{
    "user": "john doe",
    "scenarioID": "essim",
    "simulationDescription": "A simple ES with one geothermal source␣
↪and a heat demand",
    "startDate": "2019-01-01T00:00:00+0100",
    "endDate": "2020-01-01T00:00:00+0100",
    "influxURL": "http://influxdb:8086",
    "esdlContents":
↪"PD94bWwgdmVyc2lvbj0nMS4wJyBlbmNvZGluZz0nVVRGLTgnPz4KPGVzZGw6RW5lcmd5U3lzdGVtIHhtbG5zO
↪"
}
```

- **Response:**
  - **CREATED (HTTP status code - 201)**
    - *status*: CREATED
    - *id*: Unique ID for this simulation run. This ID will be tagged in all simulation data from this run.

    Example:

```
{
  "status": "CREATED",
  "id": "60c0c0a84840404e8b19df9b"
}
```

  - **BAD REQUEST (HTTP status code - 400)**
    - *status*: ERROR
    - *description*: Description of the error

    Example:

```
{
  "status": "ERROR",
  "description": "Internal error: Error in Observation Manager␣
↪init: Connecting to InfluxDB @ http://non-existing-host:8088␣
↪timed out. Please check URL!"
}
```

- **SERVICE UNAVAILABLE (HTTP status code - 503)**
  - If a simulation is already running while a new one is started, HTTP status code 503 is returned with a text body `Busy`.

5.2.2 /simulation/<simulation-id>

- **HTTP Method**: GET
- **Description**: Retrieve meta-data of simulation run. This includes information provided by the user at the time of starting the simulation and some run-time data. This API call can be used to retrieve the dashboard URL as soon as a simulation is *CREATED*
- **Request Body:**
  - None
- **Response:**
  - **NOT FOUND (HTTP status code - 404)**
    - *status*: ERROR
    - *description*: Description of the error

    Example:

```
{
    "status": "ERROR",
    "description": "SimulationID 60c0c0a84840404e8b19479b not␣
↪found!"
}
```

  - **OK (HTTP status code - 200)**
    - *status*: The last known status of the simulation.
    - *simRunDate*: The date when the simulation was run in ISO-8601 format (`YYYY-MM-DDTHH:mm:ss±hh:mm`).
    - *transport*: An HTML visualisation of ESSIM's internal transport network trees created per commodity. The HTML pages are URL-encoded and tagged with the name of the network.
    - *dashboardURL*: A link to the Grafana dashboard created by ESSIM for this simulation.

    Example:

```
{
    "esdlContents":
↪"PD94bWwgdmVyc2lvbj0nMS4wJyBlbmNvZGluZz0nVVRGLTgnPz4KPGVzZGw6RW5lcmd5U3lzdGVtIHht
↪",
    "user": "john doe",
    "scenarioID": "essim",
    "simulationDescription": "A simple ES with one geothermal␣
↪source and a heat demand",
    "startDate": "2018-12-31T23:00:00+0000",
    "endDate": "2019-12-31T23:00:00+0000",
    "status": {
        "state": "COMPLETE",
        "description": "Finished in PT1.099S"
    },
```

(continues on next page)

(continued from previous page)

```
    "influxURL": "http://influxdb:8086",
    "simRunDate": "2021-06-09T13:22:48+0000",
    "transport": [
        {
            "name": "Untitled EnergySystem Heat Network 0",
            "networkHTMLDiag": "%3C%21DOCTYPE+html%3E%0A
↪%3Chtml+lang%3D%22en%22%3E%0A++%3Chead%3E%0A++++%3Cmeta+charset
↪%3D%22utf-8%22%3E%0A%0A++++%3Ctitle%3E%0A
↪%09Untitled+EnergySystem+Heat+Network+0%0A%09%3C%2Ftitle%3E%0A
↪%0A++++%3Cstyle%3E%0A++++%0A++++.node+%7B%0A++++++++cursor
↪%3A+pointer%3B%0A++++%7D%0A%0A++++.node+circle+%7B%0A++++++fill
↪%3A+%23fff%3B%0A++++++stroke%3A+steelblue%3B%0A++++++stroke-width
↪%3A+3px%3B%0A++++%7D%0A%0A++++.node+text+%7B%0A++++++font
↪%3A+12px+sans-serif%3B%0A++++%7D%0A%0A++++.link+%7B%0A++++++fill
↪%3A+none%3B%0A++++++stroke%3A+%23ccc%3B%0A++++++stroke-width
↪%3A+2px%3B%0A++++%7D%0A++++%0A++++%3C%2Fstyle%3E%0A%0A++%3C
↪%2Fhead%3E%0A%0A++%3Cbody%3E%0A%0A%3C%21--+load+the+d3.
↪js+library+--%3E+%0A%3Cscript+src%3D%22http%3A%2F%2Fd3js.org
↪%2Fd3.v3.min.js%22%3E%3C%2Fscript%3E%0A++++%0A%3Cscript%3E%0A
↪%0Avar+treeData%3D%5B%7B%22parent%22%3A%22null%22%2C%22children
↪%22%3A%5B%7B%22parent%22%3A%22GeothermalSource_b572%28PRODUCER%29
↪%22%2C%22name%22%3A%22b505c10b-bde4-4606-8d68-7f8e0188c383
↪%28CONSUMER%29%22%7D%5D%2C%22name%22%3A%22GeothermalSource_b572
↪%28PRODUCER%29%22%7D%5D%3B%0A%0A%2F
↪%2F+**************+Generate+the+tree+diagram++*****************
↪%0Avar+margin+%3D+%7Btop%3A+20%2C+right%3A+120%2C+bottom%3A+20
↪%2C+left%3A+200%7D%2C%0A++++width+%3D+1960+-+margin.right+-
↪+margin.left%2C%0A++++height+%3D+500+-+margin.top+-+margin.bottom
↪%3B%0A++++%0Avar+i+%3D+0%2C%0A++++duration+%3D+750%2C%0A++++root
↪%3B%0A%0Avar+tree+%3D+d3.layout.tree%28%29%0A++++.size%28
↪%5Bheight%2C+width%5D%29%3B%0A%0Avar+diagonal+%3D+d3.svg.diagonal
↪%28%29%0A++++.projection%28function%28d%29+%7B+return+%5Bd.y
↪%2C+d.x%5D%3B+%7D%29%3B%0A%0Avar+svg+%3D+d3.select%28%22body%22
↪%29.append%28%22svg%22%29%0A++++.attr%28%22width%22%2C+width+
↪%2B+margin.right+%2B+margin.left%29%0A++++.attr%28%22height%22
↪%2C+height+%2B+margin.top+%2B+margin.bottom%29%0A++.append%28%22g
↪%22%29%0A++++.attr%28%22transform%22%2C+%22translate%28%22+
↪%2B+margin.left+%2B+%22%2C%22+%2B+margin.top+%2B+%22%29%22%29%3B
↪%0A%0Aroot+%3D+treeData%5B0%5D%3B%0Aroot.x0+%3D+height+%2F+2%3B
↪%0Aroot.y0+%3D+0%3B%0A++%0Aupdate%28root%29%3B%0A%0Ad3.select
↪%28self.frameElement%29.style%28%22height%22%2C+%22500px%22%29%3B
↪%0A%0Afunction+update%28source%29+%7B%0A%0A++%2F
↪%2F+Compute+the+new+tree+layout.%0A++var+nodes+%3D+tree.nodes
↪%28root%29.reverse%28%29%2C%0A++++++links+%3D+tree.links%28nodes
↪%29%3B%0A%0A++%2F%2F+Normalize+for+fixed-depth.%0A++nodes.forEach
↪%28function%28d%29+%7B+d.y+%3D+d.depth+*+250%3B+%7D%29%3B%0A%0A++
↪%2F%2F+Update+the+nodes%E2%80%A6%0A++var+node+%3D+svg.selectAll
↪%28%22g.node%22%29%0A++++++.data%28nodes%2C+function%28d%29+
↪%7B+return+d.id+%7C%7C+%28d.id+%3D+%2B%2Bi%29%3B+%7D%29%3B%0A
↪%0A++%2F%2F+Enter+any+new+nodes+at+the+parent
↪%27s+previous+position.%0A++var+nodeEnter+%3D+node.enter%28%29.
↪append%28%22g%22%29%0A++++++.attr%28%22class%22%2C+%22node%22%29
↪%0A++++++.attr%28%22transform%22%2C+function%28d%29+%7B+return+
↪%22translate%28%22+%2B+source.y0+%2B+%22%2C%22+%2B+source.x0+%2B+
↪%22%29%22%3B+%7D%29%0A++++++.on%28%22click%22%2C+click%29%3B%0A
↪%0A++nodeEnter.append%28%22circle%22%29%0A++++++.attr%28%22r%22
↪%2C+1e-6%29%0A++++++.style%28%22fill%22%2C+function%28d%29+
↪%7B+return+d._children+%3F+%22lightsteelblue%22+%3A+%22%23fff%22
↪%3B+%7D%29%3B%0A%0A++nodeEnter.append%28%22text%22%29%0A++++++.
↪attr%28%22x%22%2C+function%28d%29+%7B+return+d.children+%7C%7C+d.
↪_children+%3F+-13+%3A+13%3B+%7D%29%0A++++++.attr%28%22dy%22%2C+
↪%22.35em%22%29%0A++++++.attr%28%22text-anchor%22%2C+function%28d
↪%29+%7B+return+d.children+%7C%7C+d._children+%3F+%22end%22+%3A+
↪%22start%22%3B+%7D%29%0A++++++.text%28function%28d%29+
```

(continues on next page)

(continued from previous page)

```
        }
    ],
    "dashboardURL": "https://essim-dashboard.hesi.energy/d/
↪gUtvSu6Mk/untitled-energysystem-john-doe-2021-06-09t13-22-48-55"
}
```

5.2.3 /simulation/<simulation-id>/status

- **HTTP Method**: GET
- **Description**: Retrieve status of a simulation run. This API can be used as soon as a simulation is *CREATED*
- **Request Body:**
  - None
- **Response:**
  - **NOT FOUND (HTTP status code - 404)**
    - *status*: ERROR
    - *description*: Description of the error

    Example:

```
{
    "status": "ERROR",
    "description": "SimulationID 60c0c0a84840404e8b19479b not␣
↪found!"
}
```

  - **OK (HTTP status code - 200)**
    - **Running**
      - *State*: RUNNING
      - *Description*: Percentage progress of the simulation as a limiting to 1.0

```
{
    "State": "RUNNING"
    "Description": "0.7604529616724739",
}
```

    - **Finished**
      - *State*: COMPLETE
      - *Description*: Time for simulation to complete

```
{
    "State": "COMPLETE"
    "Description": "Finished in PT35.306S",
}
```

    - **Error**
      - *State*: ERROR
      - *Description*: Description of the error

```
{
    "State": "ERROR"
    "Description": "Cannot connect to InfluxDB service at [http://non-existing-host:8086] to query profile with id 3499a337-1785-4601-8098-3c46b6d42b7c. Please verify the URL!",
}
```

5.3 Code Example

Following is a simple code example to access the ESSIM APIs using python. Before you run this example:

1. Copy the ESDL file below the python example to the same folder as the script. Adjust the name of the file in the script (line 21) appropriately if the name of the ESDL file is changed.
2. Install the necessary python libraries using `pip install requests pytz`.
3. Start ESSIM following the instructions here.

5.3.1 Python Code:

```
1 import json
2 import time
3 import pytz
4 import base64
5 import requests
6 from os import path
7 from datetime import datetime as dt
8
9 # Common Constants
10 ESSIM_URL = 'http://localhost:8112/essim/simulation'
11 INFLUXDB_URL = 'http://influxdb:8086'
12 ESSIM_HEADERS = {'Content-Type': 'application/json', 'Accept': 'application/json'}
13 ESSIM_DATE_FORMAT = '%Y-%m-%dT%H:%M:%S%z'
14 PROGRESS_UPDATE_INTERVAL = 1  # in seconds
15
16 # Simulation-specific constants
17 ESSIM_USER = 'John Doe'
18 ESSIM_SCENARIO_ID = 'TestScenario'
19 ESDL_FILE = 'SmallestESDL_np.esdl'
20 SIMULATION_DESCRIPTION = 'Testing ESSIM API'
21 START_DATE = dt(2021, 1, 1, 0, 0, 0, 0, pytz.UTC)
22 END_DATE = dt(2022, 1, 1, 0, 0, 0, 0, pytz.UTC)
23
24
25 class ESSIMSimulation:
26
27     def __init__(self, esdl_file):
28         """
29         Constructor to create an ESSIM Simulation object
30         :param esdl_file: Path to ESDL file to simulate with ESSIM
31         """
32         self.simulation_id = None
33         self.dashboardURL = None
```

(continues on next page)

(continued from previous page)

```
        self.esdl_file = esdl_file
        if not path.exists(self.esdl_file):
            raise ValueError('File {} does not exist!'.format(path.
↪abspath(self.esdl_file)))

    def encode_esdl(self):
        """
        Function to encode contents of ESDL file into Base64
        :return: Base64-encoded contents of ESDL file
        """
        with open(self.esdl_file, 'r') as f:
            esdl_string = f.read().replace('\n', '')
        message_bytes = esdl_string.encode('utf-8')
        base64_bytes = base64.b64encode(message_bytes)
        encoded_esdl = base64_bytes.decode('utf-8')
        return encoded_esdl

    def display_dashboard_url(self):
        """
        Function to display the Grafana Dashboard URL
        """
        if self.simulation_id is None:
            print('No simulation is started yet!')
            return
        status_path = '{}/{}'.format(ESSIM_URL, self.simulation_id)
        r = requests.get(url=status_path, headers=ESSIM_HEADERS)
        response = r.json()
        status_code = r.status_code
        if status_code == 200:
            if 'dashboardURL' in response:
                self.dashboardURL = response['dashboardURL']
                print('Dashboard URL: {}'.format(self.dashboardURL))
            else:
                print('Dashboard URL not found! Simulation meta-data looks␣
↪like so:\n{}'.format(
                    json.dumps(response, indent=4, sort_keys=True)))
        elif status_code == 404:
            print(response['Description'])

    def display_progress(self):
        """
        Function to display progress of ESSIM simulation
        """
        if self.simulation_id is None:
            print('No simulation is started yet!')
            return

        status_path = '{}/{}/status'.format(ESSIM_URL, self.simulation_id)
        while True:
            r = requests.get(url=status_path, headers=ESSIM_HEADERS)
            response = r.json()
            status_code = r.status_code
            if status_code == 200:
                if response['State'] == 'RUNNING':
                    print('{:.1f}% complete'.format(100 * float(response[
↪'Description'])))
                    time.sleep(PROGRESS_UPDATE_INTERVAL)
```

(continues on next page)

(continued from previous page)

```python
                elif response['State'] == 'COMPLETE':
                    print('Simulation {}'.format(response['Description']))
                    break
                elif response['State'] == 'ERROR':
                    print('Simulation failed because of {}'.format(response['Description']))
                    break
            elif status_code == 404:
                print(response['Description'])
                break

    def start_simulation(self):
        """
        Function to start an ESSIM simulation
        """
        while True:
            data = {
                'user': ESSIM_USER,
                'startDate': START_DATE.strftime(ESSIM_DATE_FORMAT),
                'endDate': END_DATE.strftime(ESSIM_DATE_FORMAT),
                'scenarioID': ESSIM_SCENARIO_ID,
                'simulationDescription': SIMULATION_DESCRIPTION,
                'influxURL': INFLUXDB_URL,
                'esdlContents': self.encode_esdl()
            }
            print('Starting ESSIM Simulation')
            r = requests.post(url=ESSIM_URL, data=json.dumps(data), headers=ESSIM_HEADERS)
            response = r.json()
            status_code = r.status_code
            if status_code == 201:
                self.simulation_id = response['id']
                print(
                    'Successfully started ESSIM Simulation with id {id}'.format(id=response['id']))
                break
            elif status_code == 503:
                print('The ESSIM Engine is busy. Retrying in 5 seconds...')
                time.sleep(5)
            else:
                error = response['description']
                print('ESSIM Simulation failed because: {reason}'.format(reason=error))
                break


if __name__ == '__main__':
    essim = ESSIMSimulation(ESDL_FILE)
    essim.start_simulation()
    essim.display_dashboard_url()
    essim.display_progress()
    print('Done!')
```

5.3.2 ESDL File (*sim.esdl*):


```
<?xml version='1.0' encoding='UTF-8'?>
<esdl:EnergySystem xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:esdl=
↪"http://www.tno.nl/esdl" esdlVersion="v2102" version="3" id="ee10ca5f-a08d-4201-
↪823c-fa7abcc1ba1b" name="Untitled EnergySystem" description="">
  <energySystemInformation xsi:type="esdl:EnergySystemInformation" id="751842b7-80e1-
↪4e82-a222-fdebeca8a797">
    <carriers xsi:type="esdl:Carriers" id="0a3cb696-2558-48f4-9e23-aff11e1020a4">
      <carrier xsi:type="esdl:HeatCommodity" name="Heat" id="fcd62e8f-ef7a-404c-8d3f-
↪d94e9dd48cd3" supplyTemperature="80.0" returnTemperature="40.0"/>
    </carriers>
    <quantityAndUnits xsi:type="esdl:QuantityAndUnits" id="3f1a3603-2018-4a16-97e9-
↪67eeee195a5b">
      <quantityAndUnit xsi:type="esdl:QuantityAndUnitType" physicalQuantity="ENERGY"␣
↪unit="JOULE" multiplier="GIGA" id="eb07bccb-203f-407e-af98-e687656a221d"␣
↪description="Energy in GJ"/>
    </quantityAndUnits>
  </energySystemInformation>
  <instance xsi:type="esdl:Instance" name="Untitled Instance" id="42a64855-de86-4a66-
↪b267-b00c66b43b58">
    <area xsi:type="esdl:Area" name="Untitled Area" id="7bcf5cd9-d3fe-4626-859a-
↪c16d89ddf552">
      <asset xsi:type="esdl:GeothermalSource" name="GeothermalSource_b572" id=
↪"b572d6cb-313e-489f-b489-a86bbd6ecd21">
        <geometry xsi:type="esdl:Point" lon="4.702792167663575" CRS="WGS84" lat="52.
↪12170613337859"/>
        <port xsi:type="esdl:OutPort" id="15ff7099-8b7f-46af-bcc4-63e03f17722e" name=
↪"Out" connectedTo="a09857b5-5093-499c-83ce-54ecc54a7518" carrier="fcd62e8f-ef7a-
↪404c-8d3f-d94e9dd48cd3">
          <profile xsi:type="esdl:SingleValue" value="5.0" id="a3804b62-4205-4afb-
↪bf78-60cd73d339f2">
            <profileQuantityAndUnit xsi:type="esdl:QuantityAndUnitReference"␣
↪reference="eb07bccb-203f-407e-af98-e687656a221d"/>
          </profile>
        </port>
      </asset>
      <asset xsi:type="esdl:HeatingDemand" name="HeatingDemand_b505" id="b505c10b-
↪bde4-4606-8d68-7f8e0188c383">
        <geometry xsi:type="esdl:Point" lon="4.712383747100831" CRS="WGS84" lat="52.
↪12191692831886"/>
        <port xsi:type="esdl:InPort" connectedTo="15ff7099-8b7f-46af-bcc4-63e03f17722e
↪" id="a09857b5-5093-499c-83ce-54ecc54a7518" name="In" carrier="fcd62e8f-ef7a-404c-
↪8d3f-d94e9dd48cd3">
          <profile xsi:type="esdl:SingleValue" value="5.0" id="e19ae2d6-3ff7-494c-
↪b8bf-86450d855838">
            <profileQuantityAndUnit xsi:type="esdl:QuantityAndUnitReference"␣
↪reference="eb07bccb-203f-407e-af98-e687656a221d"/>
          </profile>
        </port>
      </asset>
    </area>
  </instance>
</esdl:EnergySystem>
```

CHAPTER

SIX

EXAMPLE USE CASES

CHAPTER

SEVEN

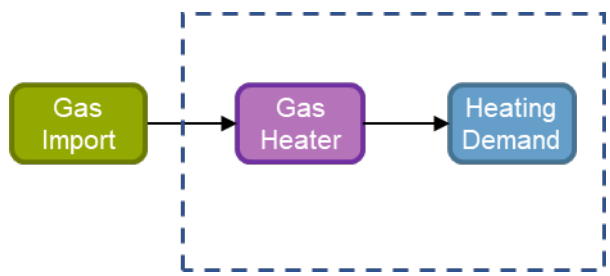
ESDL MAPEDITOR ESSIM TUTORIALS

This set of tutorials demonstrate how to use the MapEditor to create and configure energy systems and how to run and interpret basic and complex scenarios using ESSIM.

7.1 Tutorial 1: Basic Energy System

7.1.1 Description

The basic scenario models a simple *EnergySystem* consisting of a *HeatingDemand*, a *GasHeater* and an *Import* from the backbone gas network.



The dashed line indicates the *EnergySystem* boundaries. In the example above, gas is imported from an external gas source, *Import*, to the *EnergySystem* where a conversion, *GasHeater*, converts this gas to heat and supplies it to meet the *HeatingDemand*.

7.1.2 Creating an EnergySystem

To create this *EnergySystem* in MapEditor, start by creating a new ESDL file.

- Hover over the ***File*** dropdown menu and click on ***New ESDL***

Figure 1: Creating a new ESDL

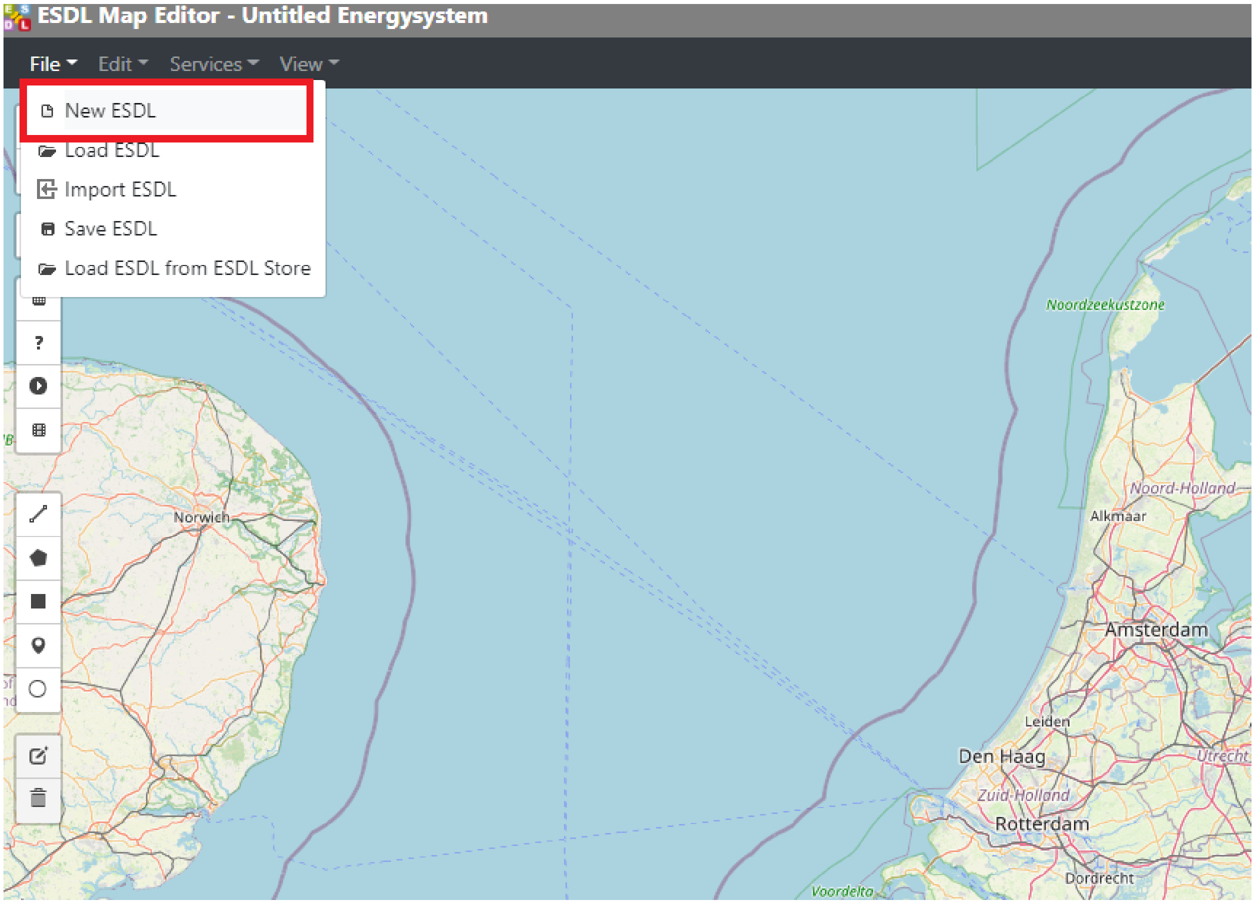
This opens up a pop-up window, where details such as *Name*, *Description*, *Email address* and *Top-level area name* can be specified. Enter the desired details as in Figure 2 (1) and click on ***Create*** (2).

Figure 2: Specifying details of a new EnergySystem

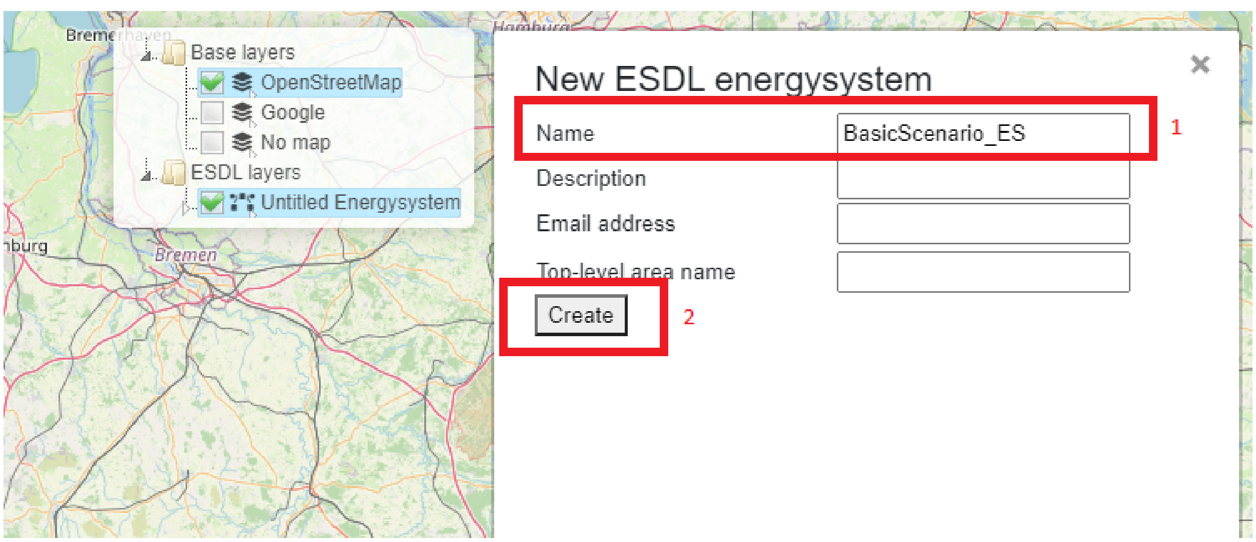
This creates a new, empty *EnergySystem*, named "BasicScenario_ES", to which ESDL elements can now be added. The elements can be added anywhere on the map, and specific locations can be found by navigating the map.

The basic *EnergySystem* in this example consists of three *EnergyAssets*: *HeatingDemand*, *GasHeater* and *Import*.

- To add each of these *EnergyAssets*, use the first dropdown menu next to ***EDR asset*** menu item.



ESDL Map Editor - Untitled Energysystem
File
Edit
Services
View
New ESDL
Load ESDL
Import ESDL
Save ESDL
Load ESDL from ESDL Store
Norwich
Noordzeekustzone
Noord-Holland
Alkmaar
Amsterdam
Leiden
Den Haag
Utrecht
Zuid-Holland
Rotterdam
Dordrecht
Voordelta



Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled Energysystem
Bremen
New ESDL energysystem
Name
BasicScenario_ES
1
Description
Email address
Top-level area name
Create
2

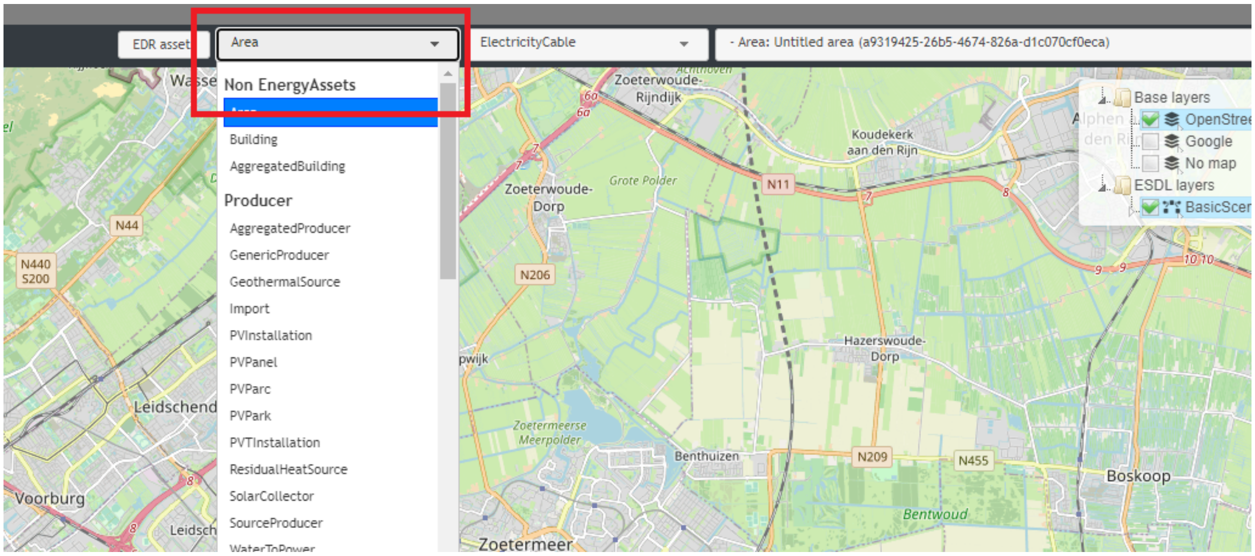


Figure 3: Selecting an EnergyAsset to create

- Clicking on the dropdown menu opens a list of Assets to choose from.
- Navigate to the *HeatingDemand* menu item and click on it.

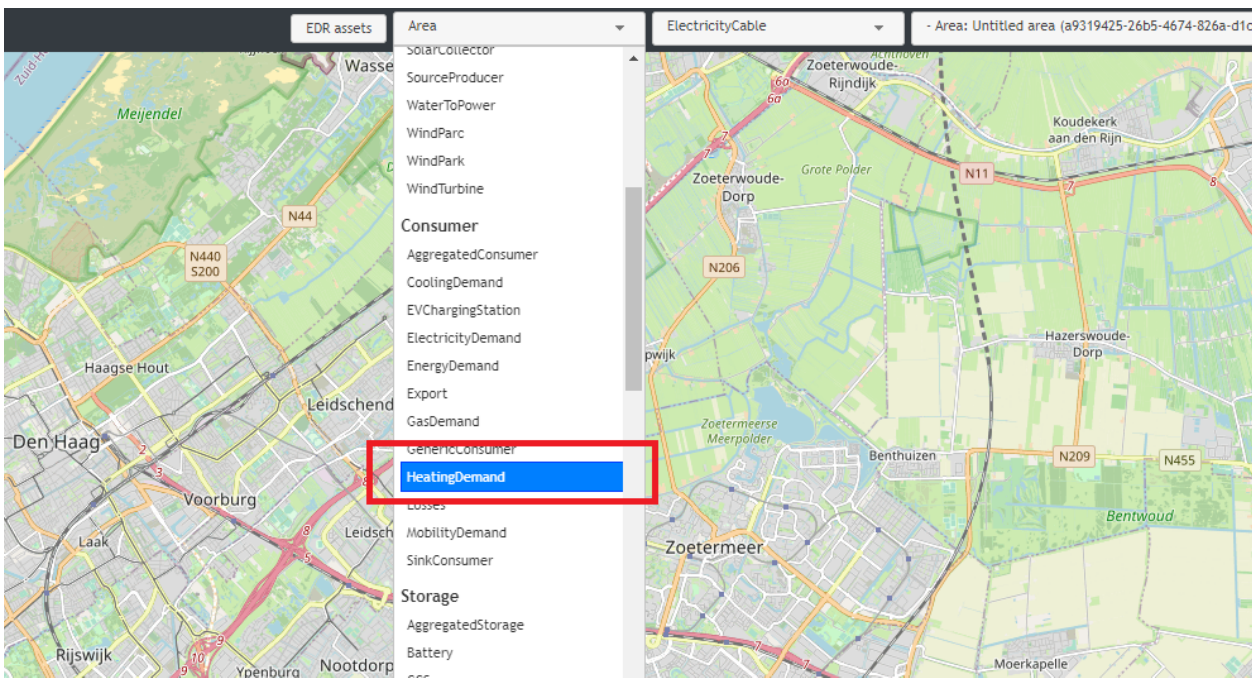


Figure 4: Selecting a new HeatingDemand to create

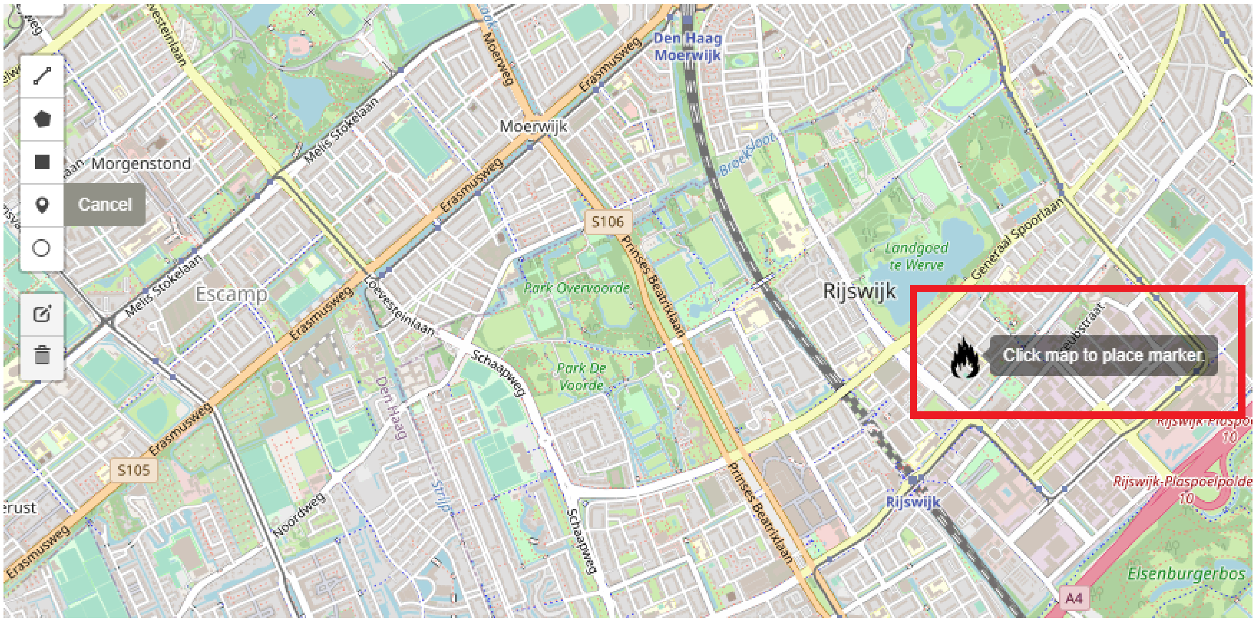
This creates a *HeatingDemand* icon next to the mouse cursor that can be placed on the map.

- Position the mouse cursor on anywhere on the map and create the *EnergyAsset* by clicking on the map.

Figure 5: Selecting a location for a new EnergyAsset

This creates a *HeatingDemand*, indicated by its icon (see the green circle in Figure 6).

- Click ***Cancel*** on the left menu bar (or press the Esc key) to complete the action.


Cancel
Click map to place marker

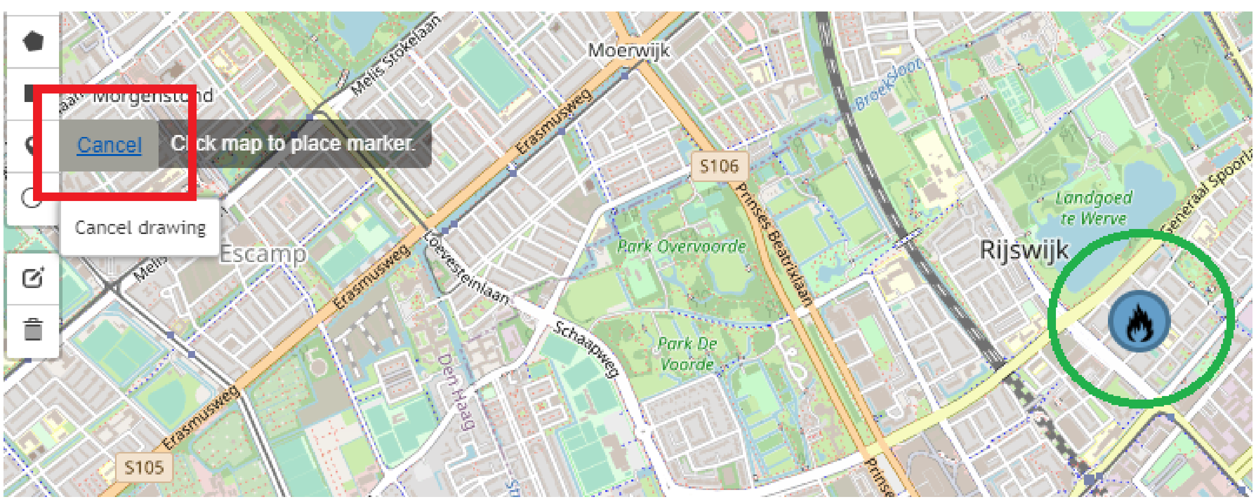

Cancel
Click map to place marker
Cancel drawing

Figure 6: Creating an EnergyAsset on the map

Follow the same steps to create *GasHeater* and *Import*. The created energy system show look as that in Figure 7

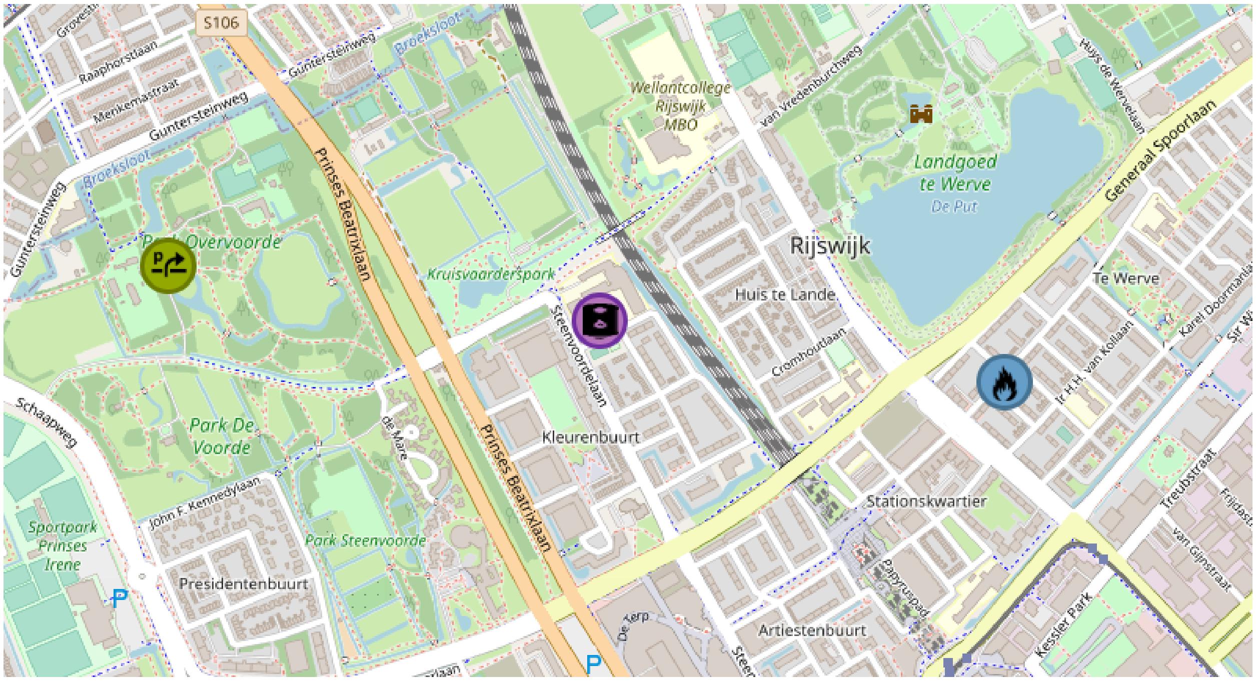


Figure 7: An EnergySystem with Import, GasHeater and HeatingDemand

The next step is to configure the created *EnergySystem* and its *EnergyAssets* by adding *Carriers* and/or *Commodities*, creating connection between the *Assets*, configuring the individual *Assets* by specifying optional and required parameters for an ESSIM simulation (such as, for example, load and production profiles, power etc.). The following subsection demonstrates how to do this.

7.1.3 Configuring an EnergySystem

To run an ESSIM simulation with a created *EnergySystem*, a number of elements have to be configured. These include:

1. Connecting *Import*, *GasHeater* and *HeatingDemand*
2. Adding Gas and Heat energy *Commodities* and assigning them to *Import* (Gas) and *HeatingDemand* (Heat) *EnergyAssets*
3. Configuring the necessary parameters of *Import*, *GasHeater* and *HeatingDemand* (e.g. specifying power, efficiency, production type, name etc.)
4. Setting load profile of *HeatingDemand*

1. Connecting *Import*, *GasHeater* and *HeatingDemand*

The first step in configuring an *EnergySystem* is to connect the created *EnergyAssets*. To connect the *Import* with *GasHeater*, follow the steps indicated in Figure 8:

- Mouseover the *Import* icon pops up a red square next to it, indicating an *OutPort*.
- Click on the red square (Indicated as 1 in the left figure) and move the mouse to the *GasHeater* icon. A dashed line appears following the cursor.
- Click on the blue square which appears next to *GasHeater* (indicating its *InPort*) (Indicated as 2 in the right picture). *Import* and *GasHeater* are now connected.
- Repeat the same procedure to connect *OutPort* of *GasHeater* (red square) and *InPort* of HeatingDemand (blue square).

The sequence of these actions determines which *EnergyAssets* are connected.

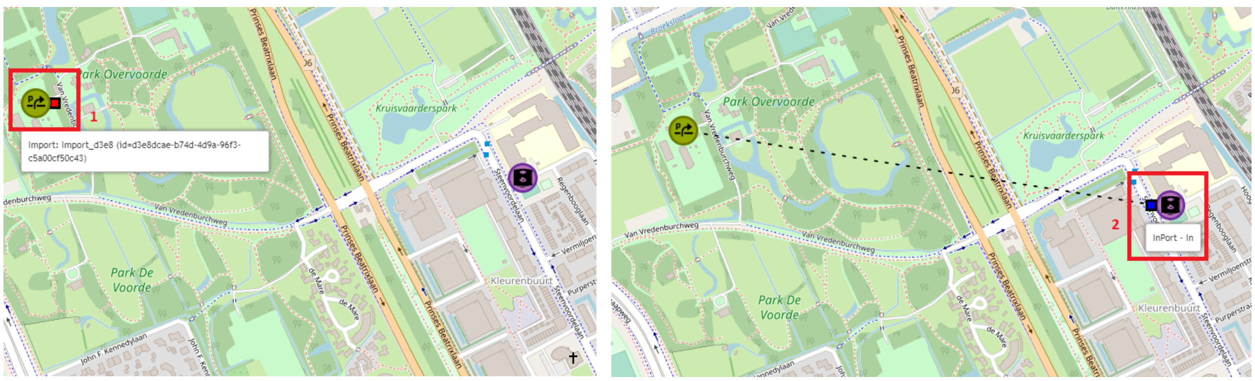


Figure 8: Connecting EnergyAssets

As a check, click on an asset and examine the information below 'Connections' in the popup window. For the *GasHeater*, the *InPort* should be connected to *Import*, and the*OutPort* should be connected to *HeatingDemand*, as shown in Figure 9. Check the connections of other two *EnergyAssets* in a similar way.

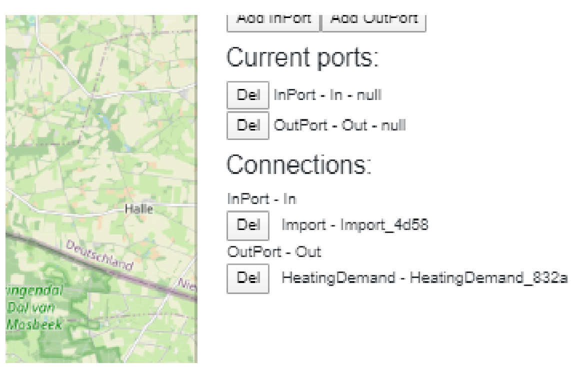
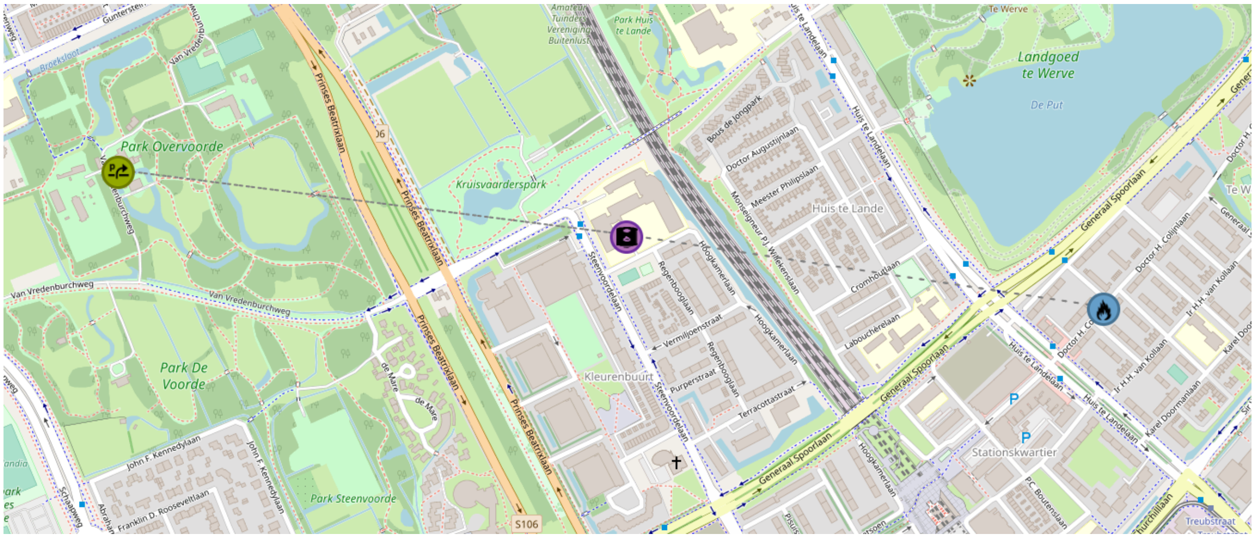


Figure 9: Verifying EnergyAsset connections

Figure 10: Connected EnergyAssets



2. Adding Gas and Heat energy *Commodities* and assigning them to *Import* (Gas) and *HeatingDemand* (Heat) *EnergyAssets*

The next step is to add energy *Carriers* and *Commodities*. In this basic scenario, two Commodities exist: gas (imported from an external system) and heat (as a result of *GasHeater* conversion). To add an energy *Commodity*:

- Mouseover the ***Edit*** menu item and click on ***Energy carriers*** (see Figure 11).

A pop-up menu opens with a list of energy *Carriers* and *Commodities* that can be added and configured.

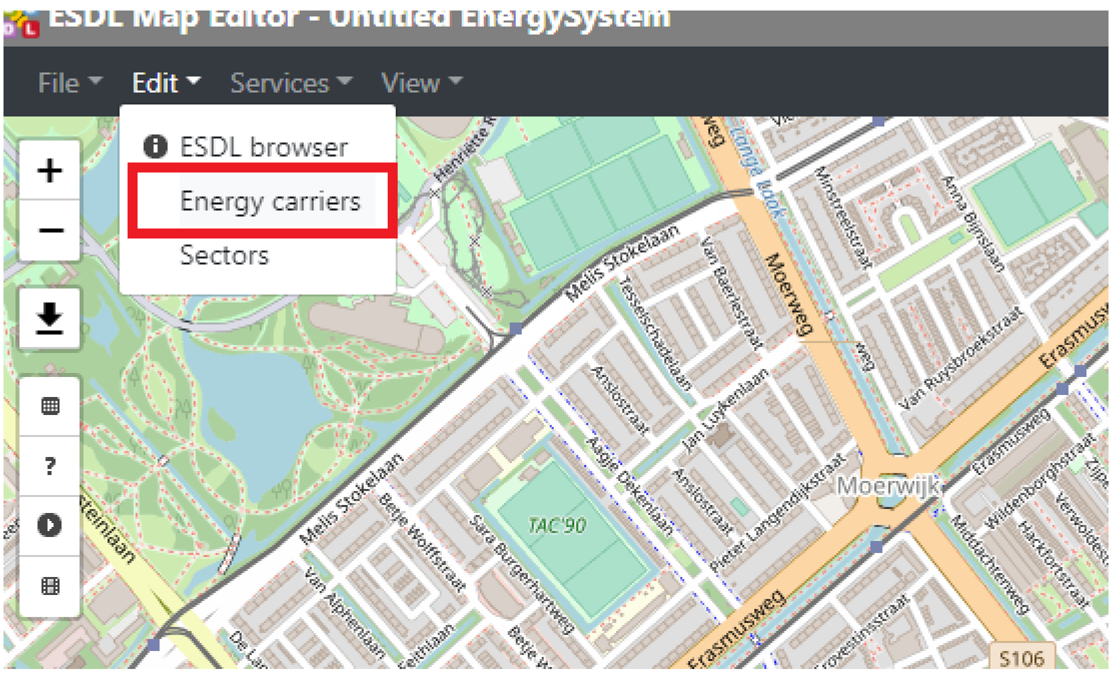


Figure 11: Adding a new Energy Commodity

- Click on the dropdown menu, and select *Gas commodity*.
- Give it a descriptive name (1) and leave the other fields blank.
- Click on ***Add carrier*** (2).

- Repeat the same process to create a *Heat commodity*.

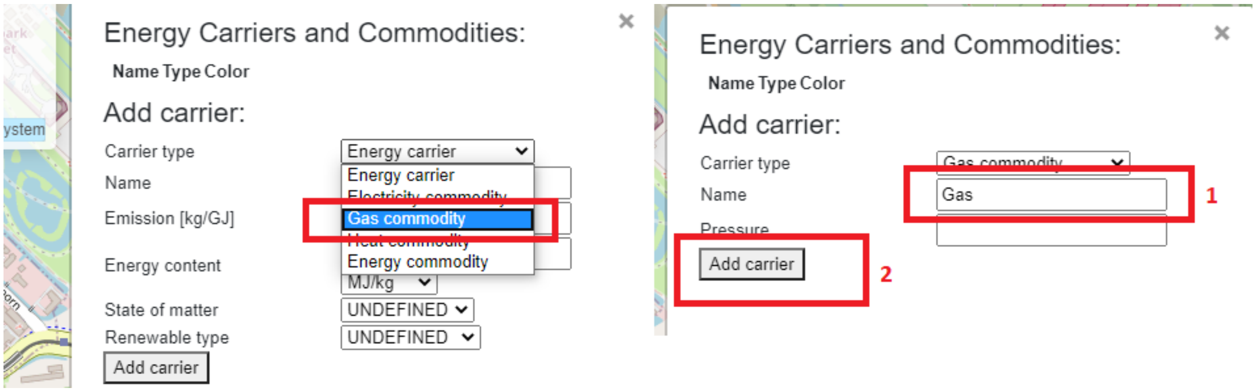


Figure 12: Adding a Gas commodity

3. Configuring the necessary parameters of Import, GasHeater and HeatingDemand (e.g. specifying power, efficiency, production type, name etc.)

After creating energy *Commodities*, assign energy *Commodities* to an *EnergyAsset* by right-clicking on its icon and selecting ***Set carrier***. In this basic scenario, energy commodities must be added to *Import* producer and *HeatingDemand* consumer. Selecting ***Set carrier*** opens a pop-up menu with a list of *Commodities* to select from.

- Right click on *Import*.
- Select ***Set carrier***.
- Choose ***Gas commodity***.

The window automatically closes and the *Commodity* is set for the selected *EnergyAsset*. Repeat the process for *HeatingDemand* and select ***Heat commodity*** from the pop-up menu.

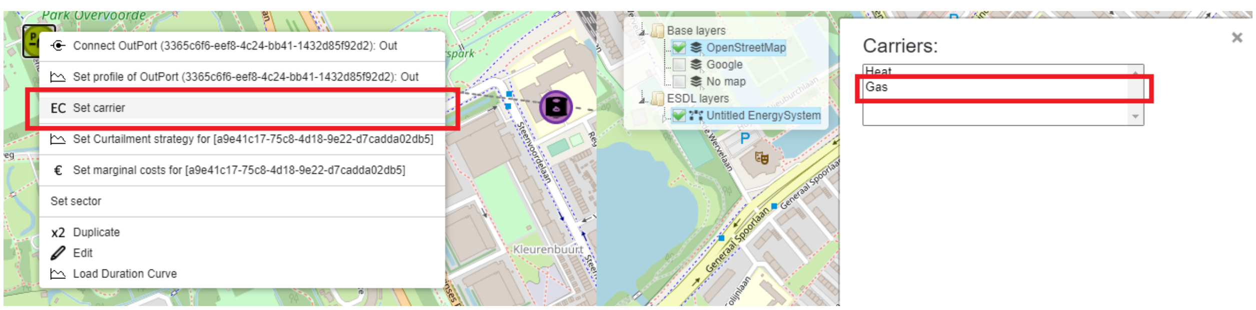
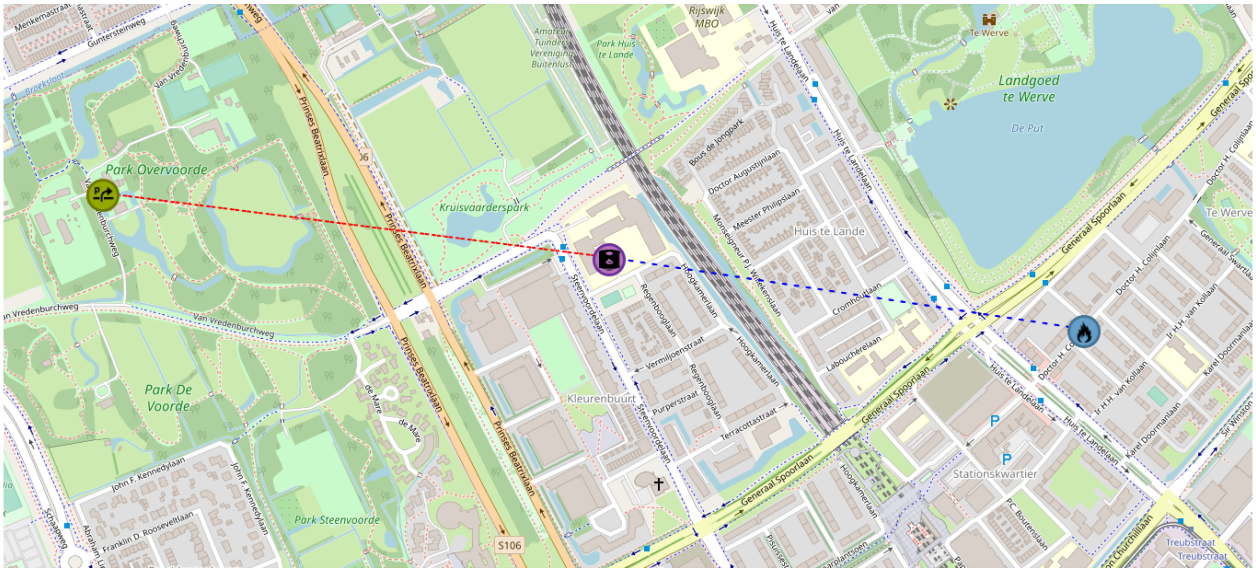


Figure 13: Setting Energy commodities

After refreshing the browser (press F5), connections between *EnergyAssets* should have different colors indicating different energy *Commodities*.

Figure 14: Colors indicating different energy Commodities



4. Configuring the necessary parameters of *Import*, *GasHeater* and *HeatingDemand*

The next step is to configure individual *EnergyAssets* by specifying required and optional parameters.

- Click on the *HeatingDemand icon* (see Figure 15) (1) . This opens up a pop-up menu with a number of configurable parameters.

In case of *HeatingDemand*, no parameters have to be changed for an ESSIM simulation.

- For the purpose of demonstration, set the **name** of the *HeatingDemand* to *HeatingDemand_Local* (2).
- Close the pop-up menu. Parameters changes are automatically saved (3).

Figure 15: Configuring HeatingDemand

Using a *GasHeater* in an ESSIM simulation requires specifying **efficiency** (percentages) and **maximum power** (in Watts).

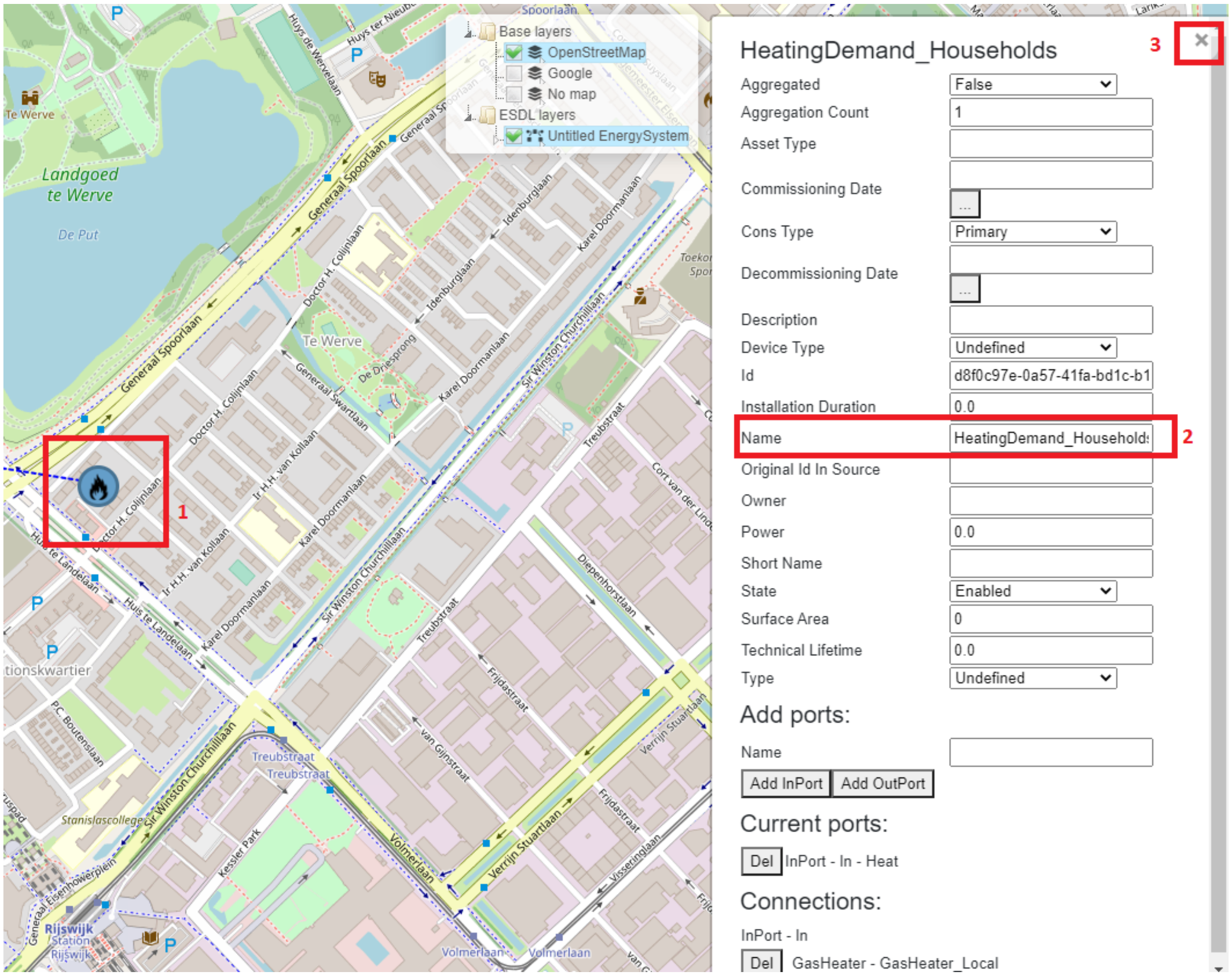
- Right click on the GasHeater icon to see the pop up window.
- Set the efficiency of the *GasHeater* to 0.9 (90%) (2).
- Set the maximum power to 6000 Watts (3) (see Figure 16).
- Close the menu (4).

Figure 16: Configuring the GasHeater

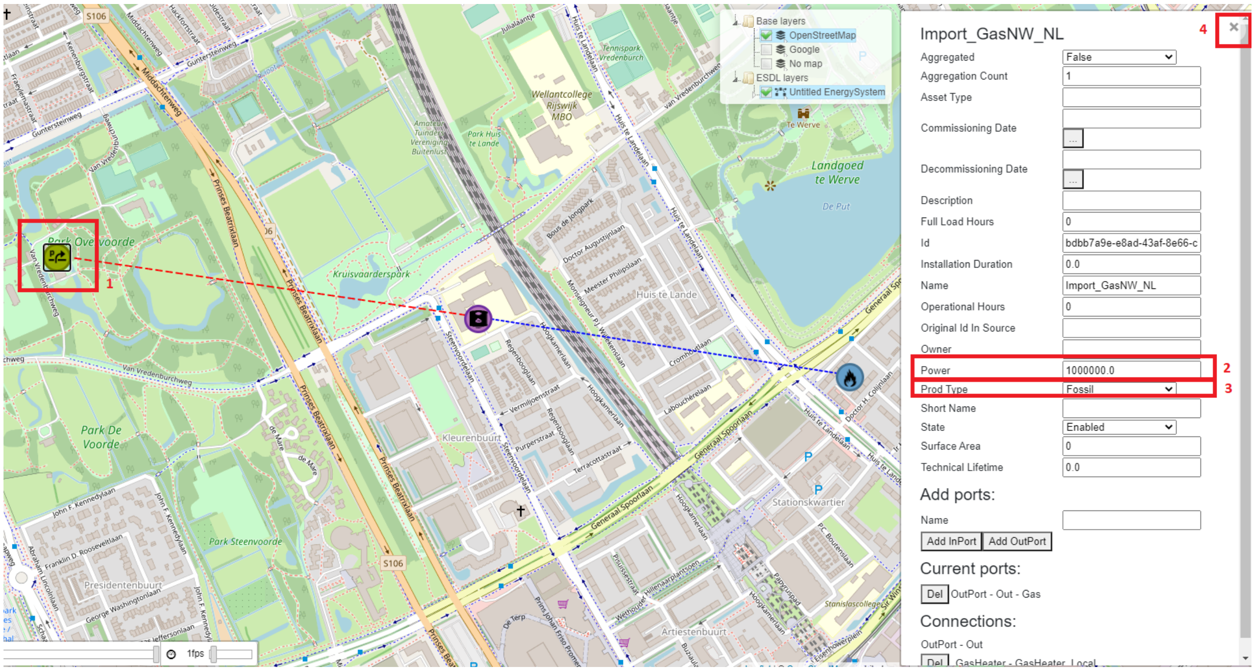
Using an Import in an ESSIM simulation requires specifying maximum power (in Watts) and production type. Follow the steps from Figure 17.

- Click on the *Import* icon (1).
- Set power to 1000000 Watts (2). Fill in only the number, not the unit.
- Set production type to *Fossil* (3)
- Close the pop-up menu (4).

Figure 17: Configuring the GasImport


Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem
HeatingDemand_Households
3
Aggregated
False
Aggregation Count
1
Asset Type
Commissioning Date
Cons Type
Primary
Decommissioning Date
Description
Device Type
Undefined
Id
d8f0c97e-0a57-41fa-bd1c-b1
Installation Duration
0.0
Name
HeatingDemand_Households
2
Original Id In Source
Owner
Power
0.0
Short Name
State
Enabled
Surface Area
0
Technical Lifetime
0.0
Type
Undefined
Add ports:
Name
Add InPort
Add OutPort
Current ports:
Del
InPort - In - Heat
Connections:
InPort - In
Del
GasHeater - GasHeater_Local
1


Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem
GasHeater_Local
4
Aggregated
False
Aggregation Count
1
Asset Type
Commissioning Date
Decommissioning Date
Description
Efficiency
0.9
2
Full Load Hours
0
Id
6db876e7-a1e7-4c09-9777-
Installation Duration
0.0
Minimum Burn Rate
0.0
Name
GasHeater_Local
Operational Hours
0
Original Id In Source
Owner
Power
6000.0
3
Short Name
State
Enabled
Surface Area
0
Technical Lifetime
0.0
Type
Undefined
Add ports:
Name
Add InPort
Add OutPort
Current ports:
Del
InPort - In - Gas
Del
OutPort - Out - Heat
Control strategy:
1
2016-01-01T00:00:00.000Z
1fps



5. Setting load profile of *HeatingDemand*

The next step in configuring an *EnergySystem* for a basic ESSIM simulation is to add load and production profiles to *EnergyAssets*. In this basic scenario, a **load** profile is set for *HeatingDemand EnergyAsset*. To add a (load) profile:

- Right-click on the *HeatingDemand* icon and select ***Set profile of InPort***.

Figure 18: Setting profiles of EnergyAssets

A pop-up window appears with a list of possible profiles to choose from. *MapEditor* enables setting a range of load and production profiles, depending on scenarios and types of *EnergyAssets*.

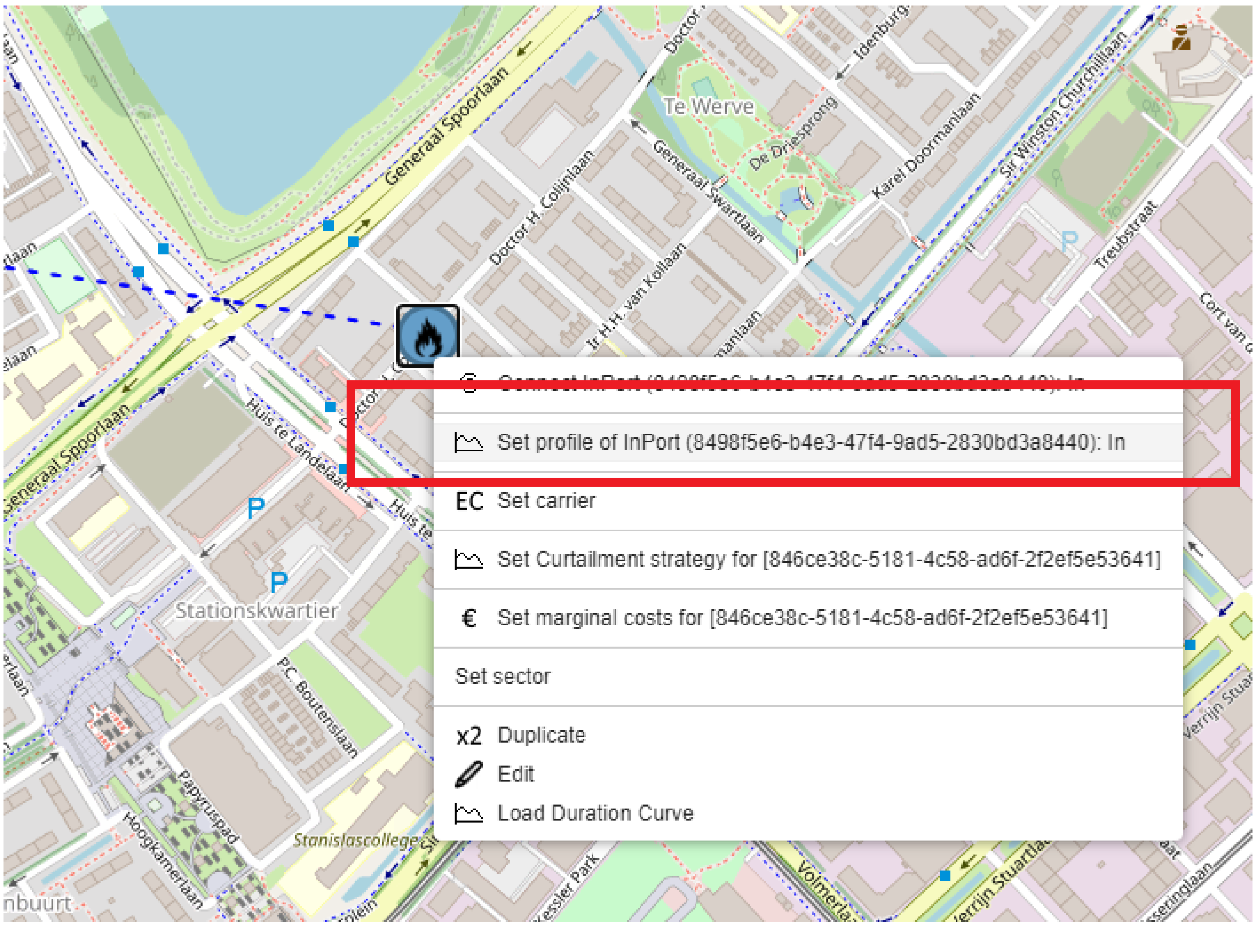
- Click on the dropdown menu (labeled ‘profile class’).
- Choose *Heating households (G1A)*, a heat demand profile of Dutch households, with hourly values.

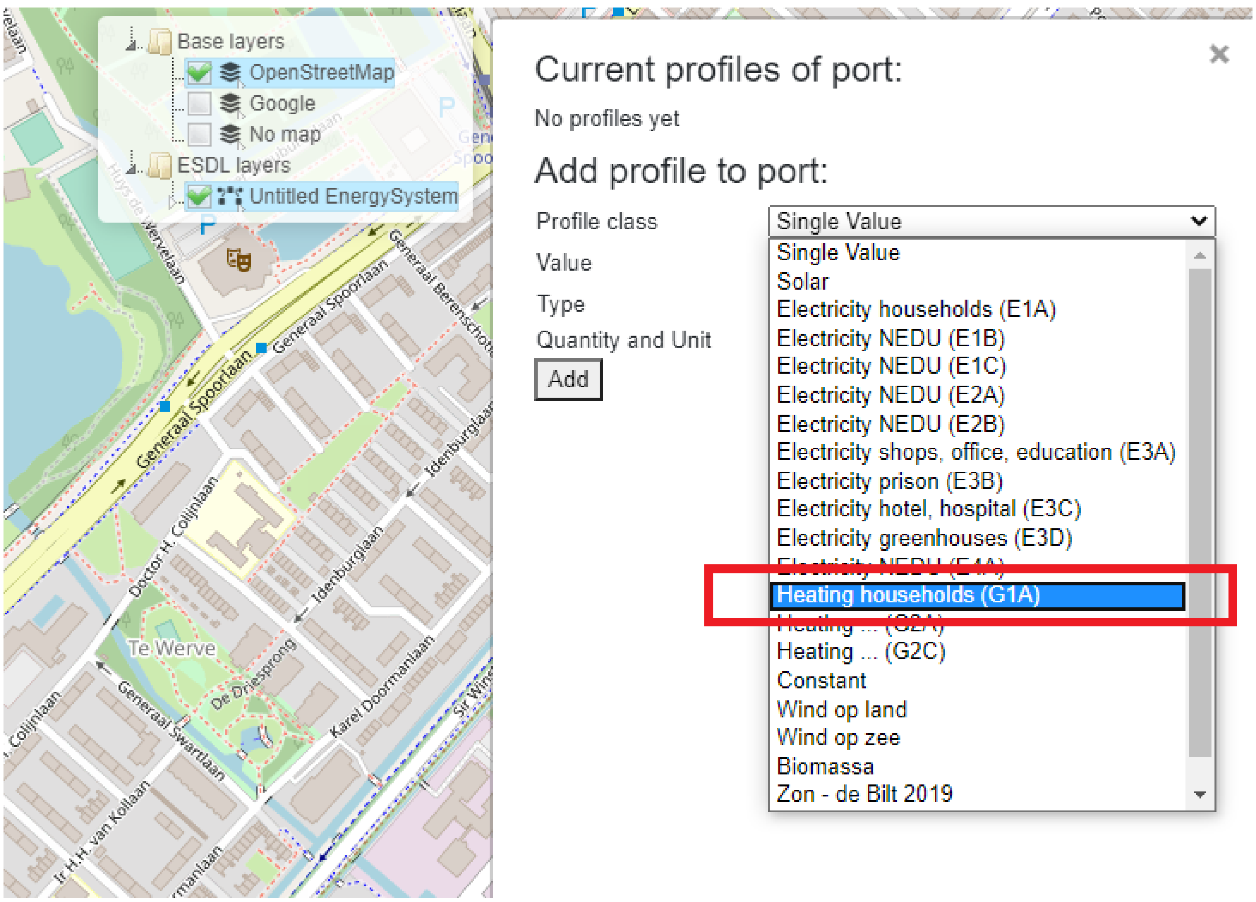
Figure 19: Choosing a load profile

As this is a normalized yearly profile, specify a *Multiplier* (1) and *Quantity and Unit* (2) for the profile. In this scenario, the yearly heating demand is 50 GJ.

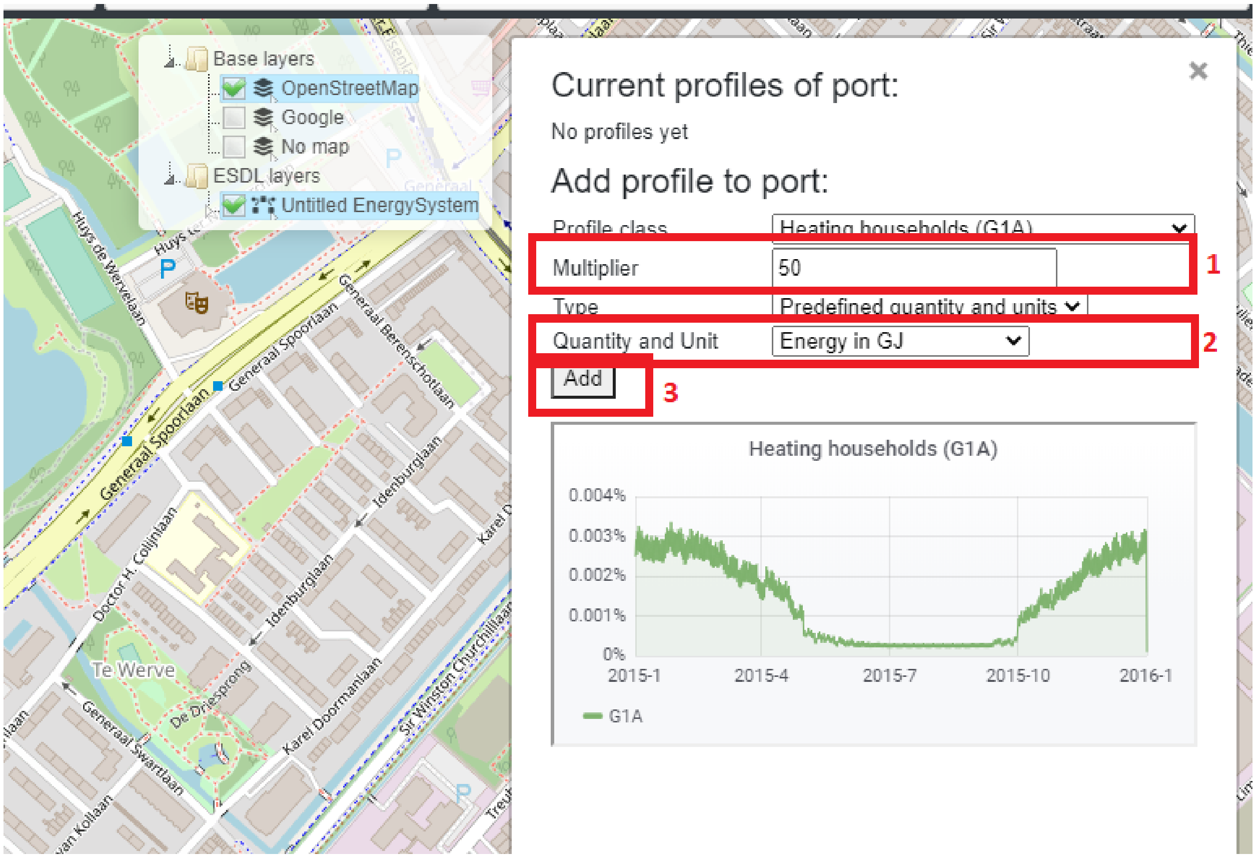
- Fill in 50 in the field ‘Multiplier’.
- Select ‘Energy in GJ’ from the dropdown list ‘Quantity and Unit’.
- Click ***Add*** (3) to close the pop-up window.

Figure 20: Setting a heating demand profile


Set profile of InPort (8498f5e6-b4e3-47f4-9ad5-2830bd3a8440): In
EC Set carrier
Set Curtailment strategy for [846ce38c-5181-4c58-ad6f-2f2ef5e53641]
€ Set marginal costs for [846ce38c-5181-4c58-ad6f-2f2ef5e53641]
Set sector
x2 Duplicate
Edit
Load Duration Curve
Generaal Spoorlaan
Te Werve
Stationskwartier



Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem
Current profiles of port:
No profiles yet
Add profile to port:
Profile class
Value
Type
Quantity and Unit
Add
Single Value
Single Value
Solar
Electricity households (E1A)
Electricity NEDU (E1B)
Electricity NEDU (E1C)
Electricity NEDU (E2A)
Electricity NEDU (E2B)
Electricity shops, office, education (E3A)
Electricity prison (E3B)
Electricity hotel, hospital (E3C)
Electricity greenhouses (E3D)
Heating households (G1A)
Heating ... (G2C)
Constant
Wind op land
Wind op zee
Biomassa
Zon - de Bilt 2019



7.1.4 Saving the model

Now is a good time to save the model. This model will be used in the next tutorials, as a base configuration.

To save the created *EnergySystem*, mouseover ***File*** menu item (1) and select ***Save ESDL*** (2) (see Figure 21). Save the file as 'Tutorial1_Scenario.esdl'.

Figure 21: Saving an EnergySystem

7.1.5 Running an ESSIM simulation

Now that all the parameters are set, an ESSIM simulation can be run for this *EnergySystem*. To run a simulation:

- Click the "Play" button on the left-hand side menu (see Figure 22).

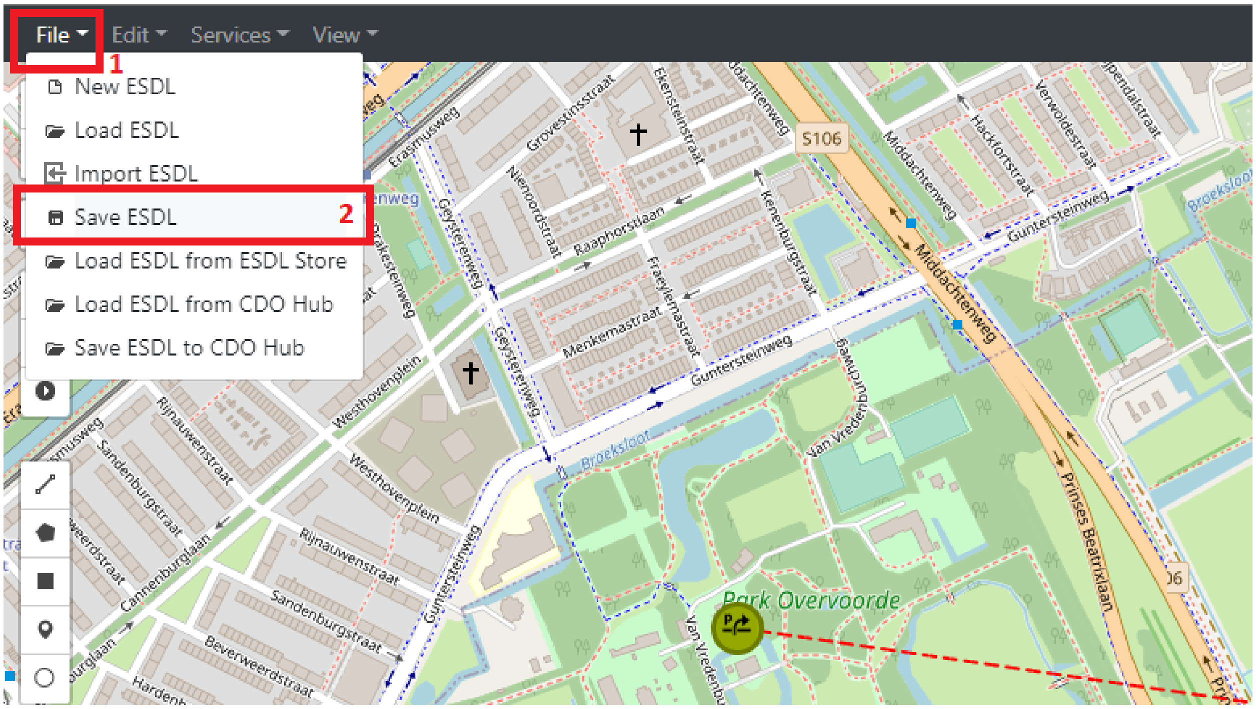
Figure 22: Running an ESSIM simulation

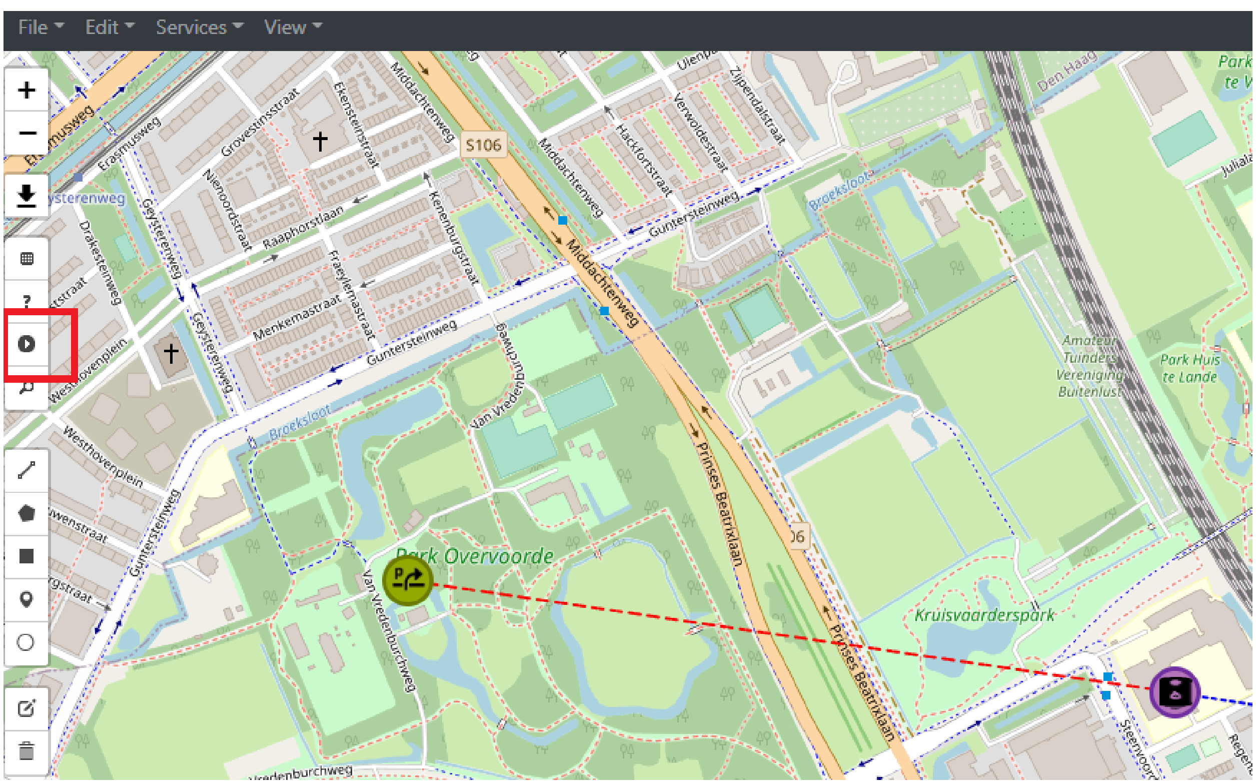
A pop-up window opens with ESSIM simulation parameters (see Figure 23).

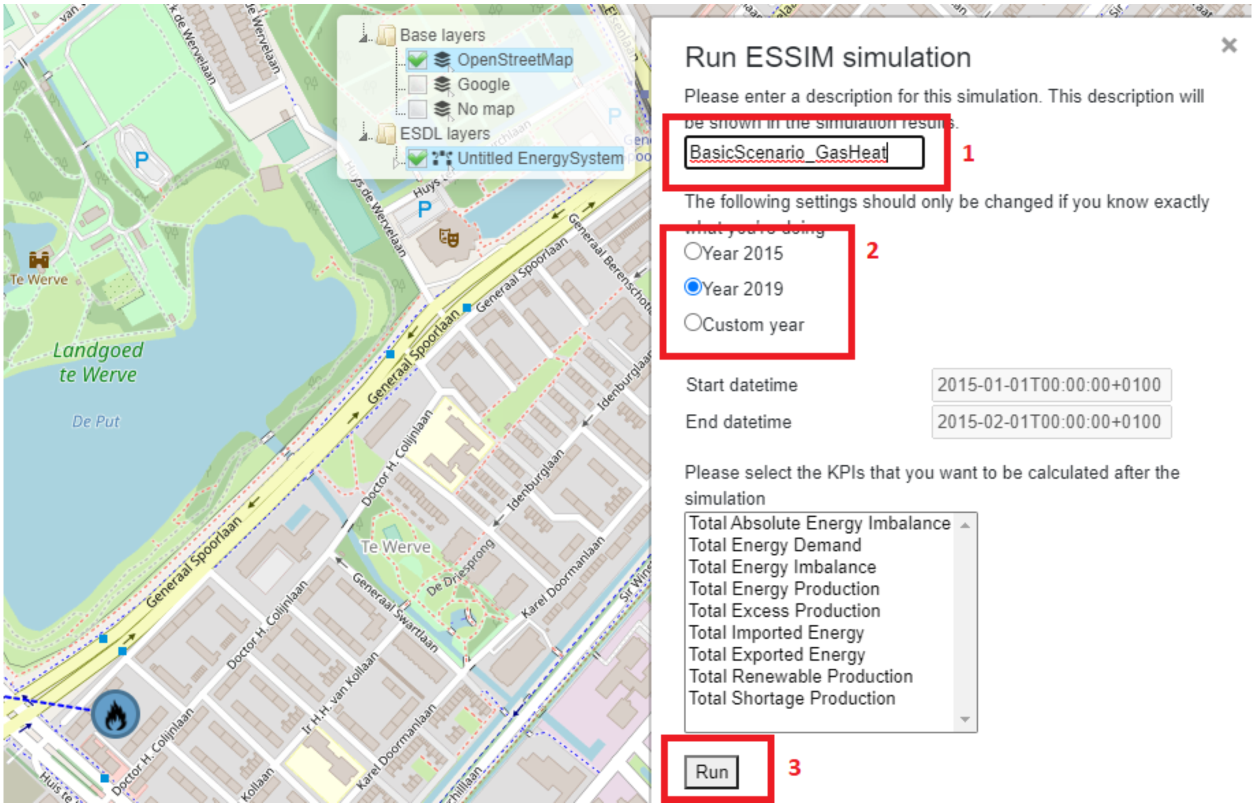
- Enter a simulation description (1).
- Choose a year or a period to simulate (2). For this example, choose Year 2019.
- Click ***Run*** (3) to run the simulation.

Figure 23: Configuring an ESSIM simulation

Once the simulation is finished, a link to a dashboard appears (see Figure 24). Clicking on the link opens a dashboard with ESSIM simulation results.


File
Edit
Services
View
1
New ESDL
Load ESDL
Import ESDL
Save ESDL
2
Load ESDL from ESDL Store
Load ESDL from CDO Hub
Save ESDL to CDO Hub
Park Overvoorde


File
Edit
Services
View
Park Overvoorde
Kruisvaarderspark


Run ESSIM simulation
Please enter a description for this simulation. This description will be shown in the simulation results.
BasicScenario_GasHeat
1
The following settings should only be changed if you know exactly what you're doing
2
Year 2015
Year 2019
Custom year
Start datetime
2015-01-01T00:00:00+0100
End datetime
2015-02-01T00:00:00+0100
Please select the KPIs that you want to be calculated after the simulation
Total Absolute Energy Imbalance
Total Energy Demand
Total Energy Imbalance
Total Energy Production
Total Excess Production
Total Imported Energy
Total Exported Energy
Total Renewable Production
Total Shortage Production
Run
3
Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem

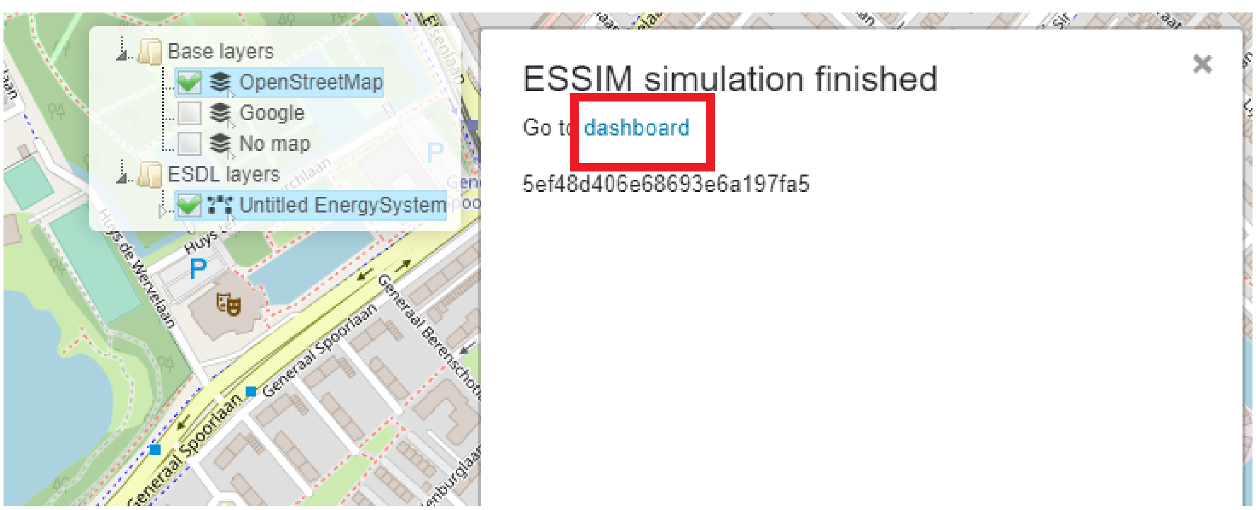

ESSIM simulation finished
Go to dashboard
5ef48d406e68693e6a197fa5
Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem

Figure 24: Navigating to ESSIM simulation results

7.1.6 Interpreting the results

ESSIM simulation results are displayed in a Grafana dashboard in a separate window. In the upper-right corner of the dashboard, Network Balances are displayed, indicating any potential imbalances. In this scenario, there are two networks: a gas network and a heat network. ESSIM displays results for each of these networks separately. Both heat and gas network are in balance, as indicated by a green **OK** flag (see Figure 25).

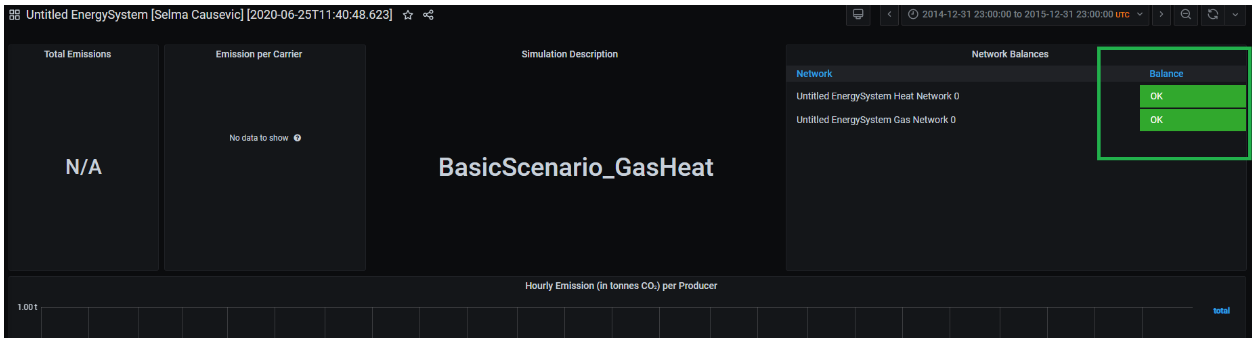


Figure 25: Network balances in an ESSIM simulation

To see load, production and imbalance in each of the networks on an hourly basis, scroll to the bottom of the page. As seen in Figure 26, results for each of the networks are displayed in a separate panel. The graphs show hourly data, while total production, demand and imbalance per energy asset is displayed on the right-hand side. Production is indicated by a negative sign, while demand is indicated by a positive sign.

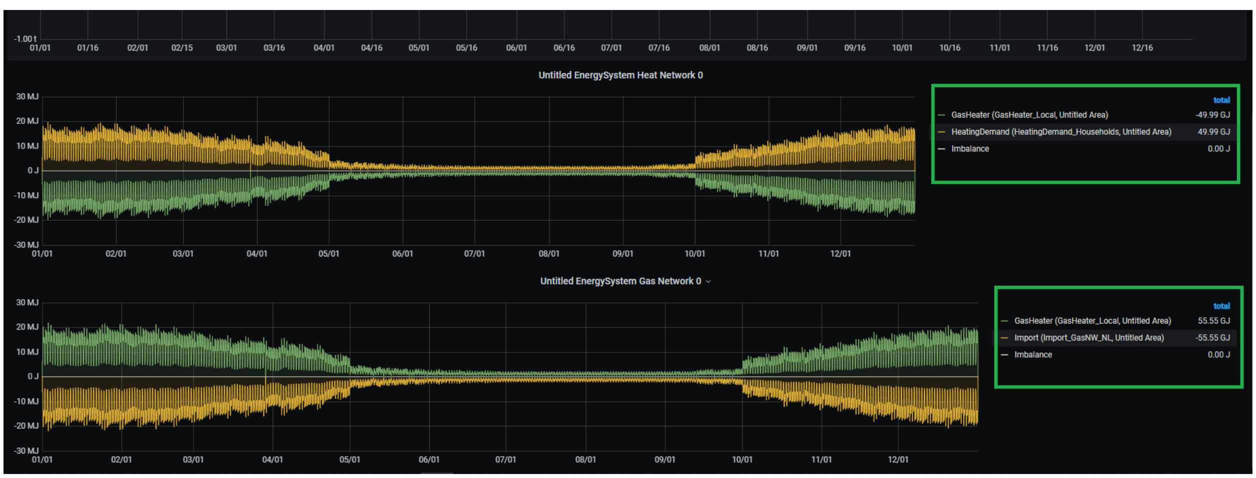
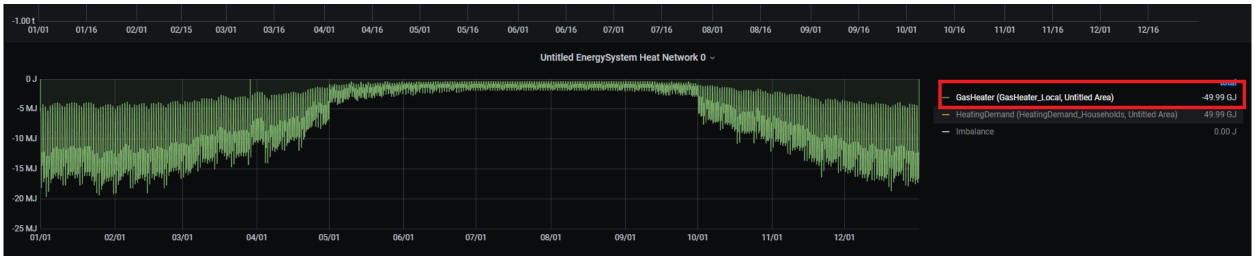


Figure 26: Load, production and imbalance in networks

To see details for a specific energy asset, for example the *GasHeater*, click on its name in the Heat Network panel (see Figure 27).

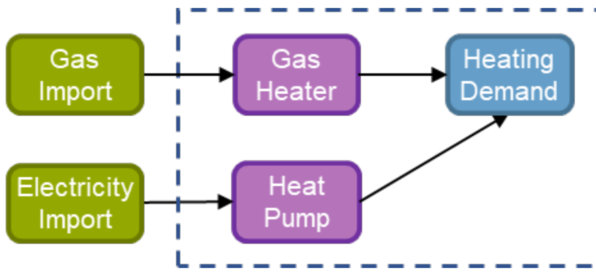
Figure 27: GasHeater production curve



7.2 Tutorial 2: Not so basic Energy System

7.2.1 Description

This tutorial demonstrates how to set preferences for specific (types of) producers in an *EnergySystem* to meet the demand of specific consumers. To do so, the *EnergySystem* from the previous tutorial is extended by adding another heat source, a *HeatPump*, which converts electricity to heat. Both the *HeatPump* and the *GasHeater* are connected to the same*HeatingDemand*. To produce heat, the *HeatPump* is connected to an external electricity grid, an *Import*, and supplies the heat to the *HeatingDemand*.



7.2.2 Loading the base configuration

To continue with the model created in the previous tutorial, load the 'Tutorial1_Scenario.esdl.

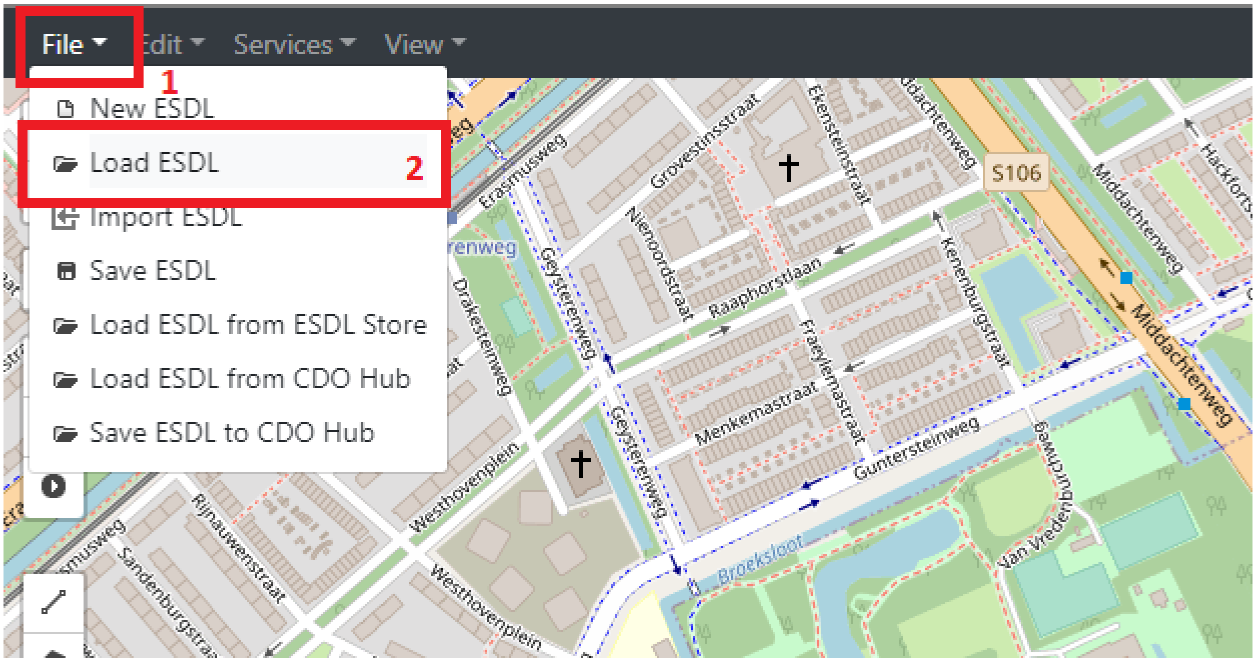
- Select ***File*** (1).
- Select ***Load ESDL*** (2) (see Figure 28).
- Navigate to the file location and load the saved .esdl file.

Figure 28: Loading an EnergySystem

7.2.3 Creating and configuring a *HeatPump* and an electricity *Import*

To extend the basic *EnergySystem* from the previous tutorial, follow these steps:

- Create new *[EnergyAssets]*
  - *Import EnergyAsset*
    - Power: 1000000 W
    - Production Type: Fossil
  - *HeatPump EnergyAsset*
    - Power: 3000 W



    * Efficiency: 1.0 (100%)
    * Coefficient of performance (COP): 3.0

- Connect the *EnergyAssets*
  - *OutPort* of the Import with the *InPort* of the *HeatPump*
  - *OutPort* of the *HeatPump* with the *InPort* of the *HeatingDemand*
- Create an electricity *Commodity*
- Assign the electricity *Commodity* to the electricity Import
- Re-assign the heat *Commodity* to the *HeatingDemand*

In this scenario, there are two heat sources supplying the *HeatingDemand*: a *GasHeater* and a *HeatPump*. In the current setting, an ESSIM simulation treats both sources equally and uses them at the same time to meet the heating demand of the consumer (see Figure 29).

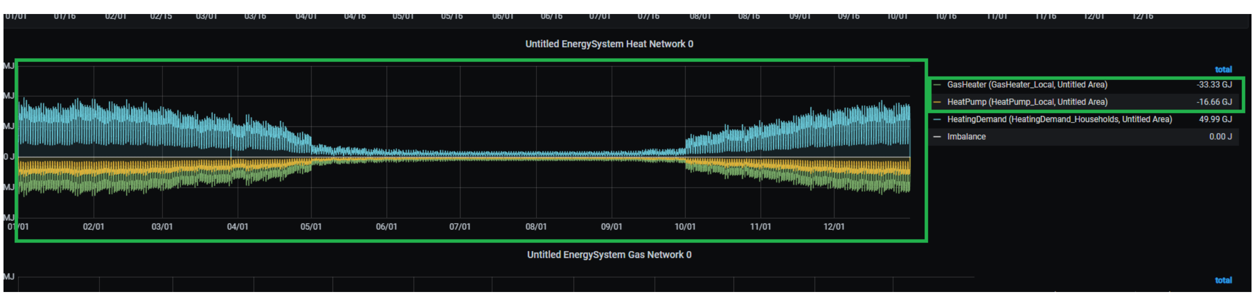


Figure 29: HeatPump and GasDemand as producers of equal priority

However, in this scenario, we would like to use the *HeatPump* to its maximum capacity at all times, and use the *GasHeater* only at times when there is not enough production from the *HeatPump*. Therefore, *HeatPump* needs to have higher priority that the *GasHeater*.

Specifying marginal costs of *HeatPump* and *GasHeater EnergyAssets*

ESSIM uses the concept of marginal costs to determine the priority of *EnergyAssets* (or the order in which EnergyAssets are used). In ESSIM they are specified in relative terms to each other, not in absolute terms. They can have a minimum value of 0 (the cheapest producer) and a maximum value of 1 (the most expensive producer). To meet an energy demand, ESSIM first uses the cheapest producer (the highest priority) up to its maximum power. Therefore, to determine the order in which heat sources are used, marginal costs have to be set for *GasHeater* and *HeatPump*. Since we want to first use the *HeatPump* at all times possible, it will have lower marginal costs compared to that of the *GasHeater*.

To set the marginal costs of the *HeatPump*;

- Right-click on its icon (1) and select ***Set marginal costs*** (2) (see Figure 30).

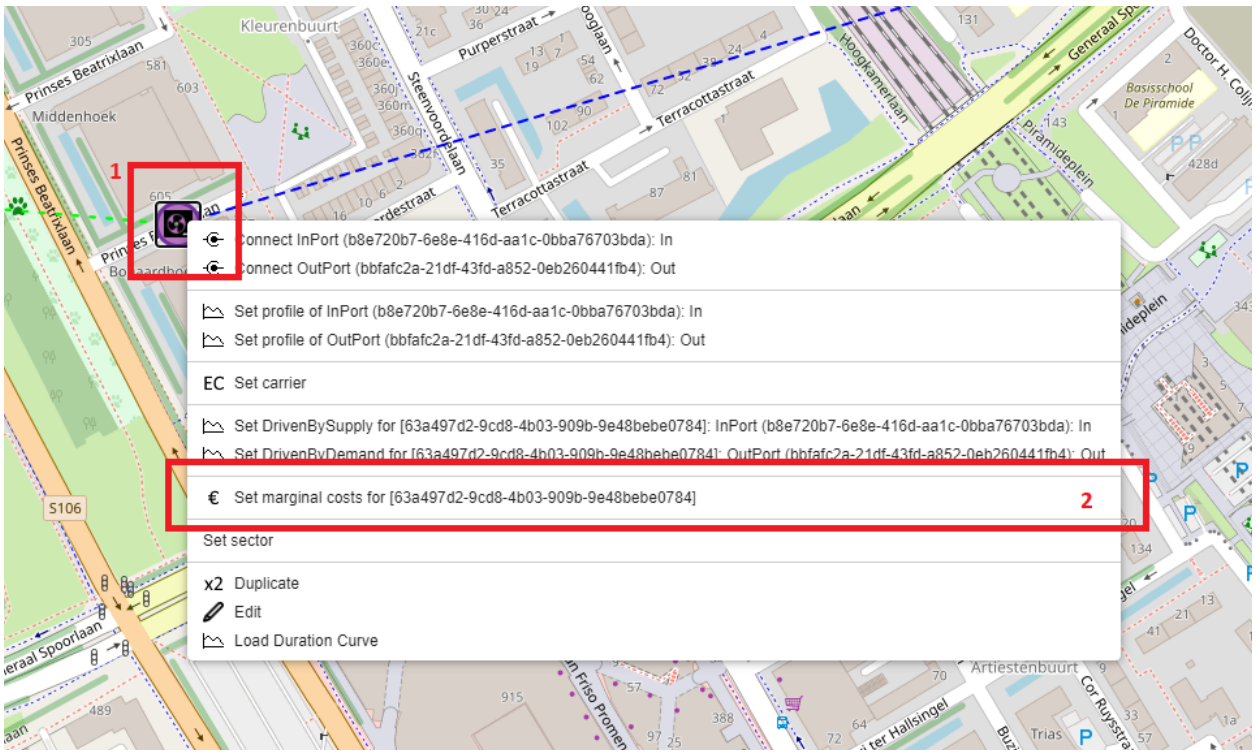


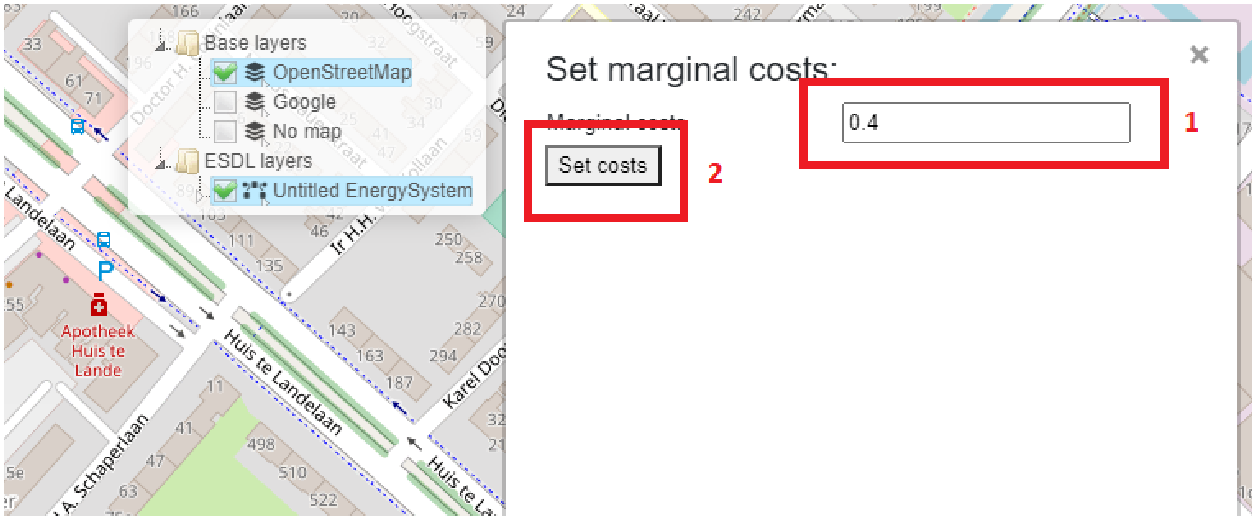
Figure 30: Setting marginal costs of the HeatPump

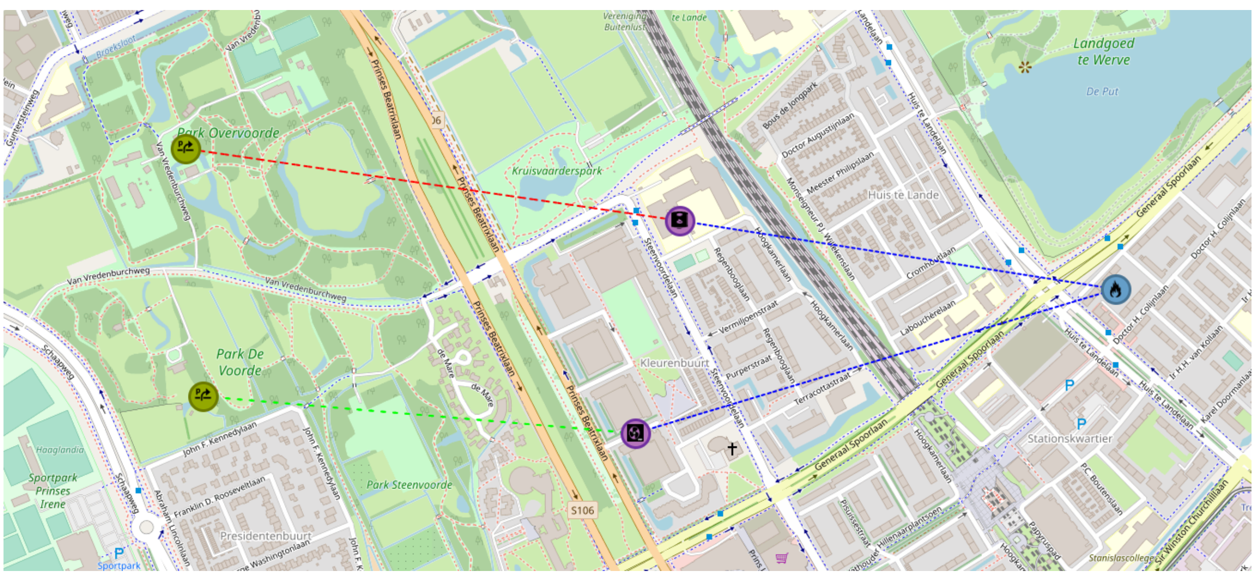
- Set the costs to 0.4 (1).
- Click ***Set costs*** to save changes and close the window (2) (see Figure 31).

Figure 31: Setting marginal costs of the HeatPump

- Repeat the same procedure for the *GasHeater*, but set its costs to 0.6.

The priorities of heat producers are now set. Marginal costs can be changed or checked following the same procedure of setting the initial values. The *EnergySystem* scenario can now be simulated.


Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem
Set marginal costs:
0.4
Set costs
1
2
Apotheek Huis te Lande
Huis te Landelaan


Park Overvoorde
Park De Voorde
Landgoed te Werve
Kruisvaarderspark
Park Steenvoorde
Kleurenbuurt
Presidentenbuurt
Stationskwartier
Huis te Lande

7.2.4 Running an ESSIM simulation and interpreting the results

Run the simulation as shown in Section Tutorial 1: Basic Energy System, Running an ESSIM simulation. ESSIM results now show three networks, namely Gas, Heat and Electricity.

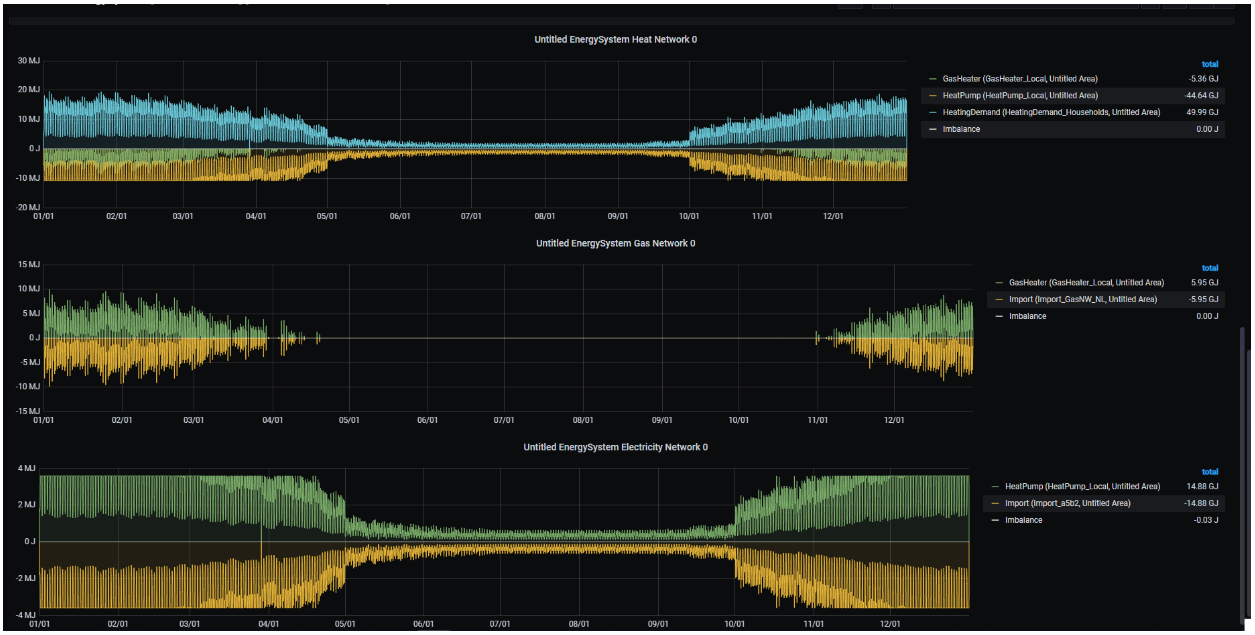


Figure 32: ESSIM simulation results

To explore the effect of prioritization by setting marginal costs, observe the Heat Network panel.

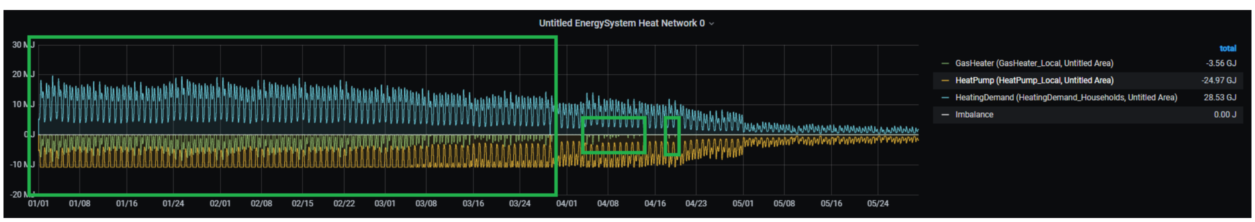


Figure 33: Heat Network with producer priorities

Compared to Figure 27, where the *HeatPump* and the *GasHeater* are producing at the same time, Figure 31 shows that the *GasHeater* is producing only at times when there is not enough production from the *HeatPump* to meet all the demand. This is better illustrated if we zoom-in into a specific period. To zoom-in, simply click and drag the mouse over the desired period. As seen in Figure 34, the *GasHeater* supplies heat only at specific intervals of time (see the green graph).

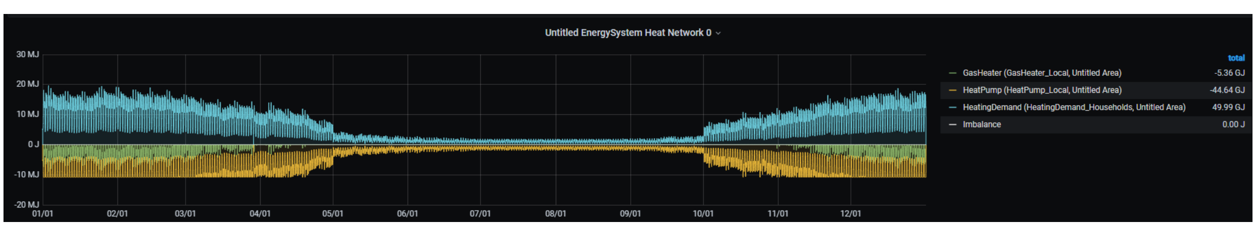


Figure 34: HeatPump as the highest priority producer

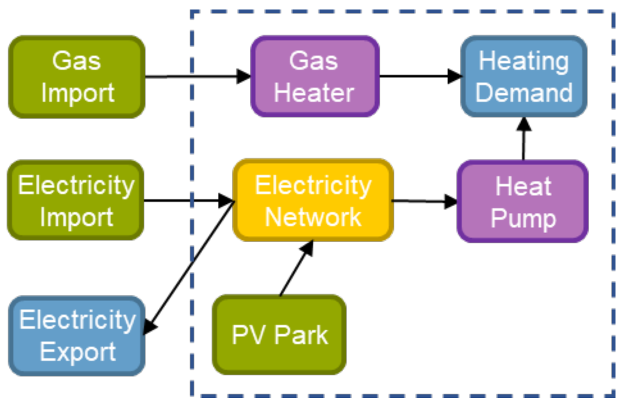
7.2.5 Saving the model

To save the model, follow the steps from Tutorial 1: Basic Energy System, Saving the model. Name the model 'Tutorial2_Scenario.esdl'.

7.3 Tutorial 3: Renewable source export excess

7.3.1 Description

This tutorial demonstrates a scenario where overproduction from a local renewable resource (a PV park) is exported to the backbone electricity grid. The *PVPark* and the electricity *Import* are connected to a local *ElectricityNetwork*, which supplies electricity to the *HeatPump*.



7.3.2 Load the model

To build upon the previously created *EnergySystem*, load the 'Tutorial2_Scenario.esdl' file from Tutorial 2: Not so basic Energy System. To load the model, follow the steps from Tutorial 2: Not so basic Energy System, Loading the base configuration.

7.3.3 Creating and Configuring the *ElectricityNetwork*, *PVPark* and *Export*

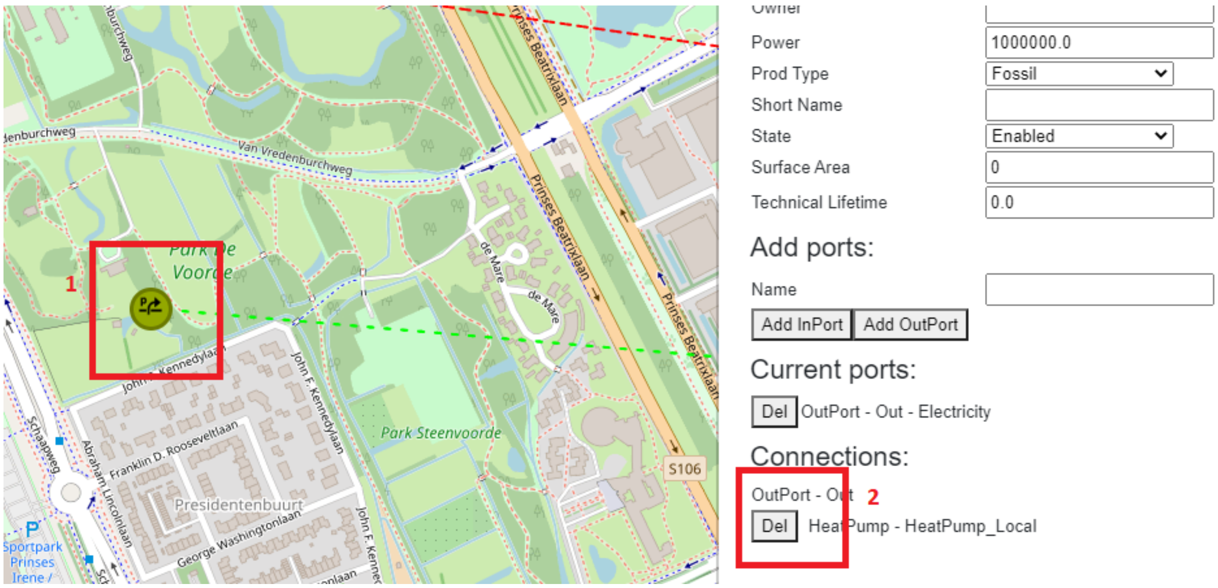
In this tutorial, a local *PVPark* is added as a local electricity source. As the *HeatPump* is the only electricity consumer in this scenario, both *PVPark* and Import are producing to meet this demand. To connect both the *PVPark* and the Import to the *HeatPump*, this tutorial creates a local *ElectricityNetwork* to which all electricity consumers and producers are connected.

To do so, the Import and the *HeatPump* from Tutorial 2 first have to be disconnected. To remove the connection between the assets, follow the steps indicated in Figure 35:

- Click on the *Import* asset (1).
- Scroll to the bottom of the pop-up menu where *Asset* connections are listed.
- Click on ***Del*** *OutPort* connection to the *HeatPump* (2).

The *Import* and the *HeatPump* and now disconnected.

Figure 35: Removing connections between EnergyAssets



The next step is to create a local *ElectricityNetwork* and connect the Import and the *HeatPump* to this network. To create an *ElectricityNetwork EnergyAsset*, follow the steps from Tutorial 1: Basic Energy System.

Next, a *PVPark EnergyAsset* is created. Creating a *PVPark* differs from the previously introduced *EnergyAssets*. To create a *PVPark*, follow the steps indicated in Figure 36:

- Select the *PVPark* item from the dropdown menu (1).
- Click on the map to position it (2).

As indicated by a pentagon shape on the left-hand menu (see the green mark), a *PVPark* is a polygon shape, requiring multiple points to be drawn.

Figure 36: Creating a PVPark EnergyAsset

- Continue drawing the desired shape of the PVPark by clicking on the map to create its vertices (1) (see Figure 37).
- Select ***Finish*** to finish the shape (2).

Figure 37: Creating a polygon shape of the PVPark

Figure 38 shows the created *PVPark EnergyAsset*.

Figure 38: The created PVPark EnergyAsset

Next, add a production profile to the *PVPark*.

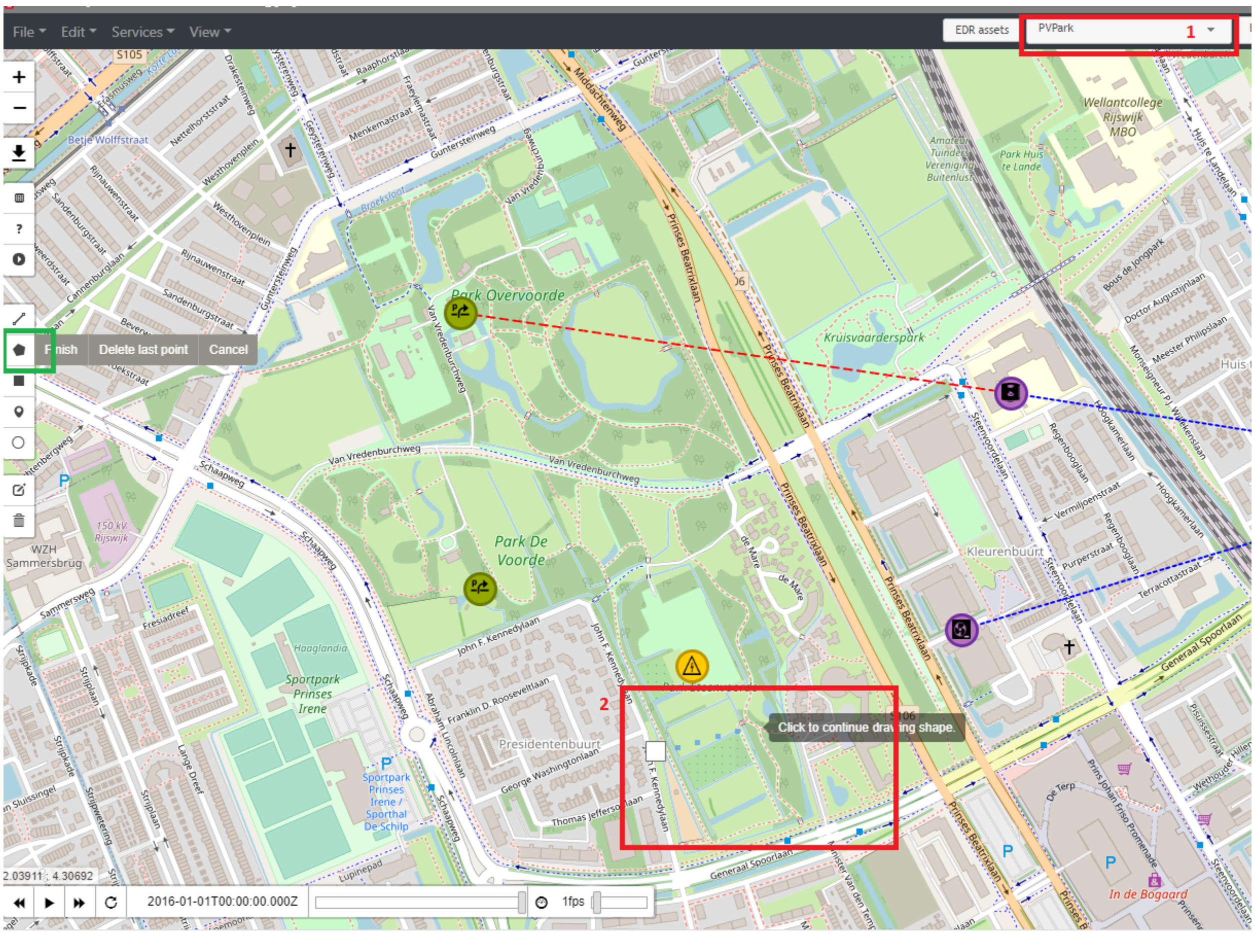
- Right-click on the *PVPark* icon.
- Choose *Solar* from the drop-down menu of the *Profile class* (1), a normalized solar production profile on an hourly basis.

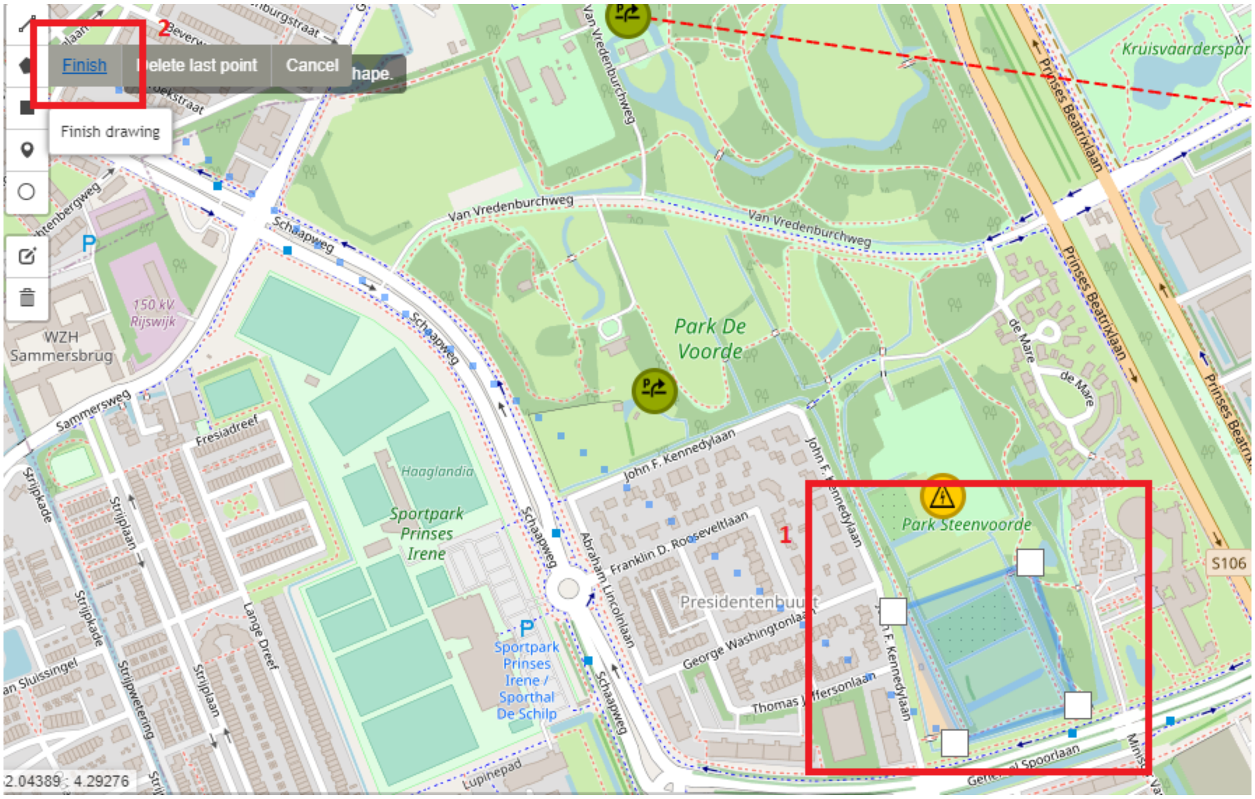
In this scenario, the yearly production of this *PVPark* is 25 GJ.

- Enter 25 for the *Multiplier*.
- Leave *Energy in GJ* as *Quantity and Unit*.
- Click on ***Add*** (3) to close the pop-up window and save the profile (see Figure 39).

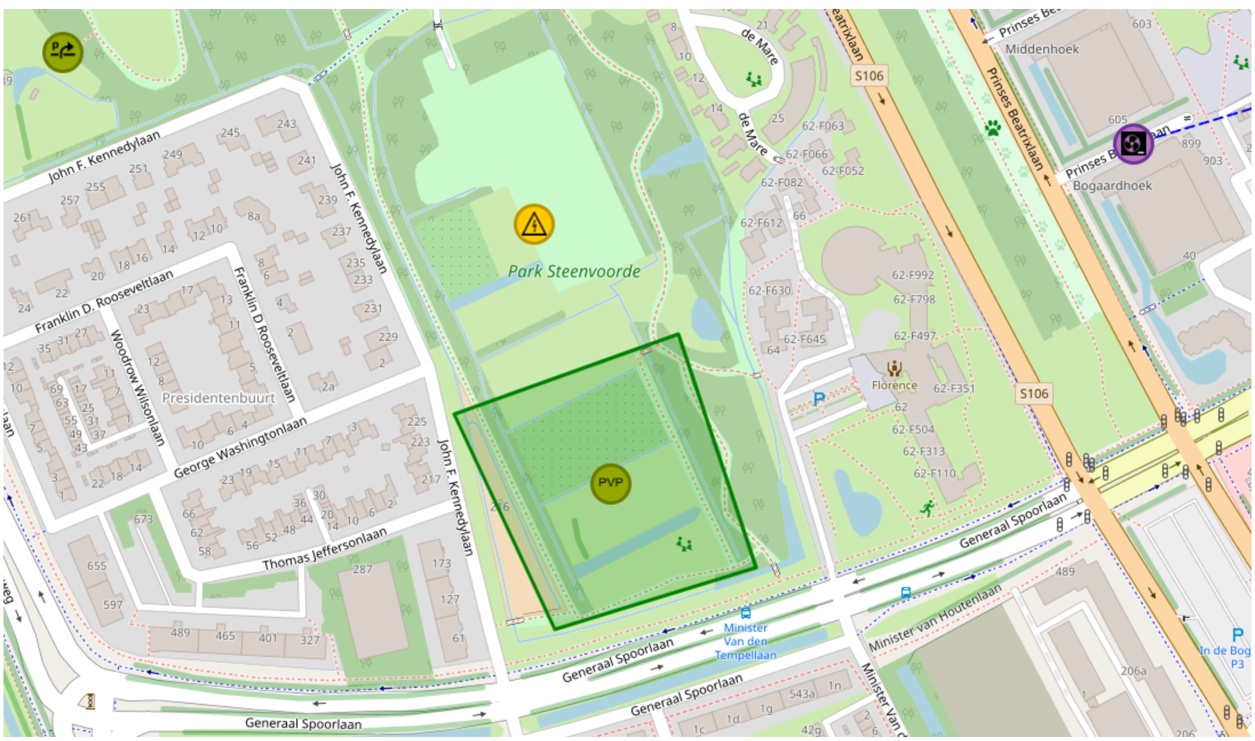
Figure 39: Setting the production profile of the PVPark

- To change any other *PVPark* parameter (e.g. the name), click on its icon and enter the desired details.

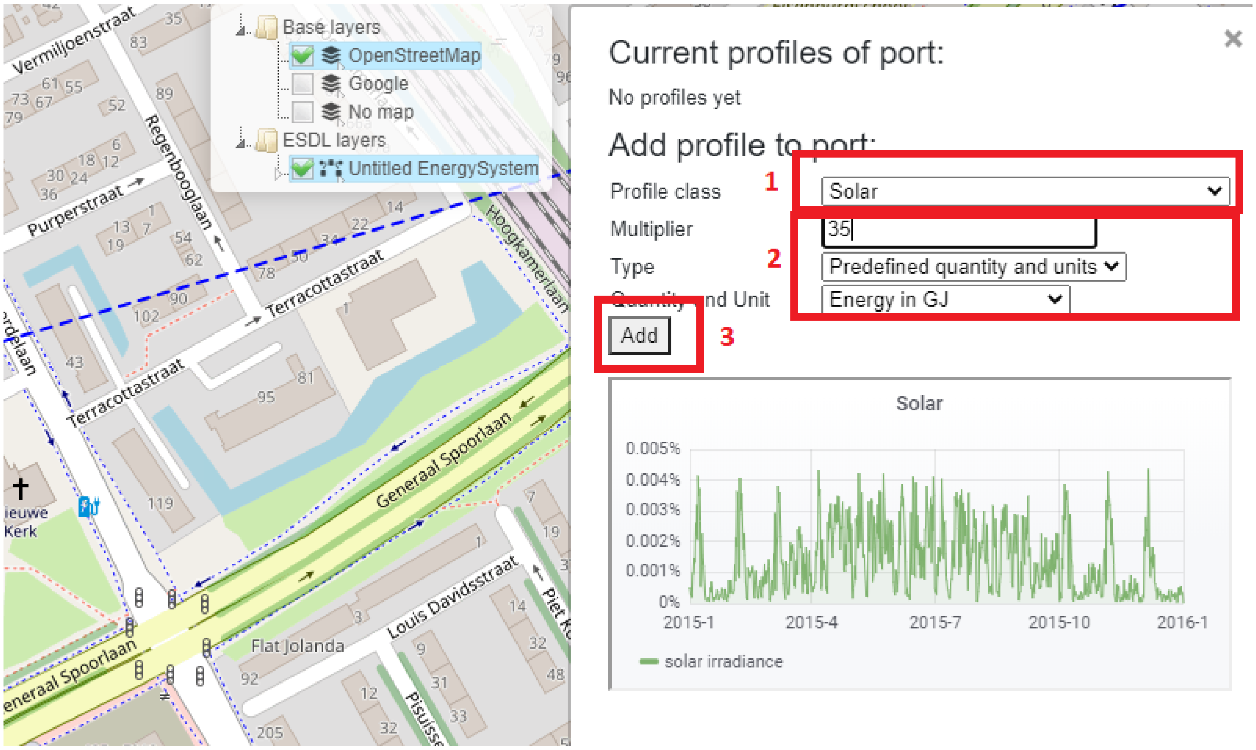

File
Edit
Services
View
EDR assets
PVPark
1
Finish
Delete last point
Cancel
Park Overvoorde
Kruisvaarderspark
Park De Voorde
Kleurenbuurt
Presidentenbuurt
2
Click to continue drawing shape.
2016-01-01T00:00:00.000Z
1fps



Finish
Delete last point
Cancel
Finish drawing
1
2
Park De Voorde
Park Steenvoorde
Sportpark Prinses Irene
Presidentenbuurt



Park Steenvoorde
PVP
Presidentenbuurt
Generaal Spoorlaan



- Make sure that *Renewable* is selected as *Production Type*.

Figure 40: Configuring the parameters of the PVPark

The next step is to connect the *Import*, the *PVPark* and the *HeatPump* to the *ElectricityNetwork,* and assign energy *Carriers*. To do so, follow the steps from Tutorial 1: Basic Energy System:

- Connect the *OutPort* of the Import to the *InPort* of the *ElectricityNetwork.*
- Connect the *OutPort* of the *PVPark* to the *Inport* of the *ElectricityNetwork.*
- Connect the *OutPort* of the *ElectricityNetwork* to the *InPort* of the *HeatPump.*
- Assign electricity *Commodity* to the *ElectricityNetwork.*
- Refresh the browser to see the changes.
- Check the connections by selecting an *EnergyAsset* and looking at its 'Connections' in the pop-up menu.

Figure 41 shows the created *EnergySystem* with a *PVPark*.

Figure 41: EnergySystem with a PVPark

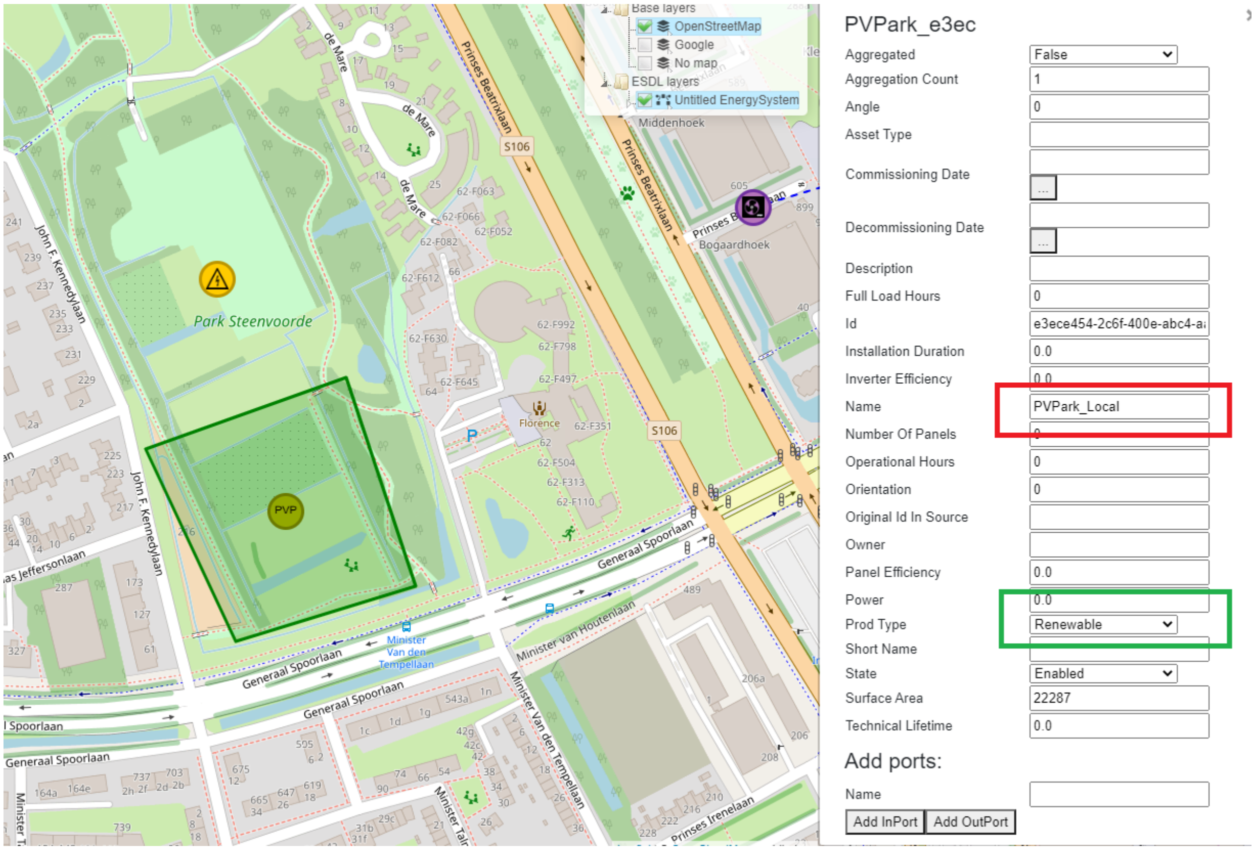
7.3.4 Running and interpreting an ESSIM simulation

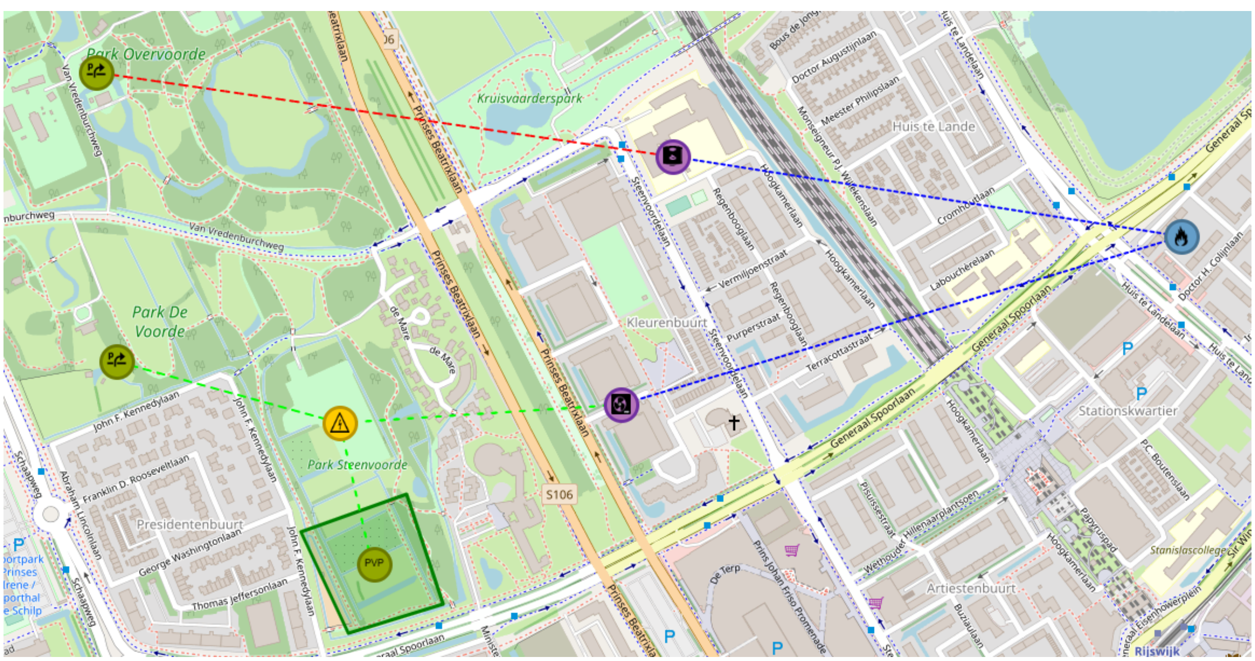
Running an ESSIM simulation for the created scenario generates an imbalance in the *ElectricityNetwork* (see Figure 42). Positive imbalance indicates that there is overproduction from the *PVPark,* causing the system failure.

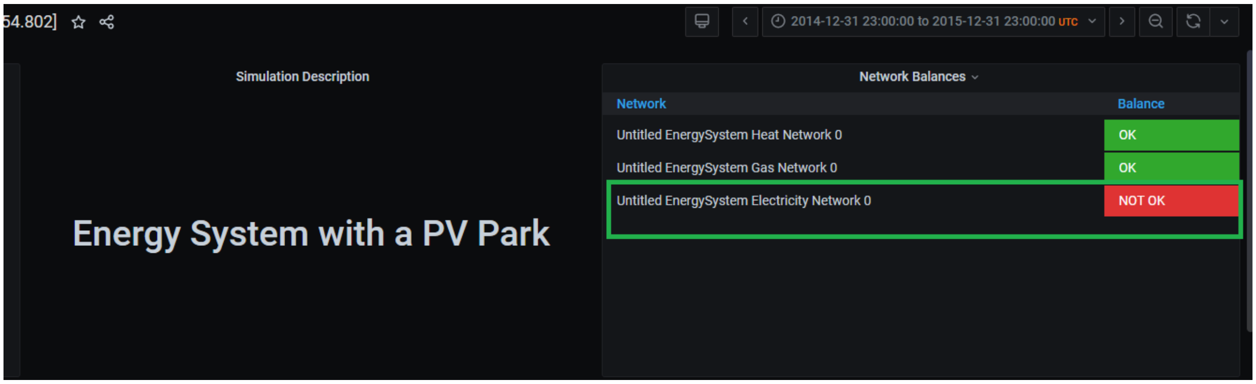
Figure 42: ElectricityNetwork imbalance

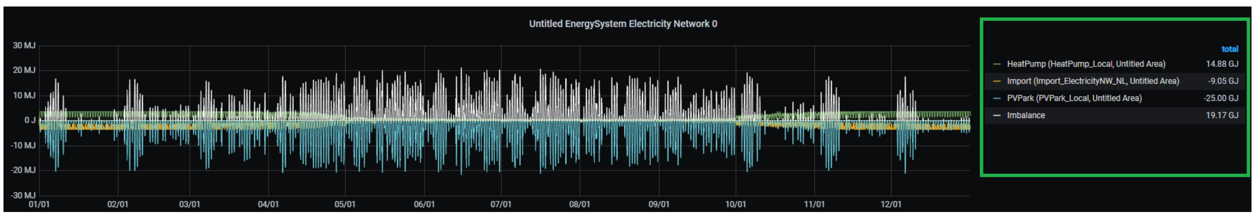
To prevent the system failure, overproduction generated by the *PVPark* can be exported to an external electricity consumer (the backbone gird, for example). To simulate export to the backbone grid, create an *Export EnergyAsset* by following the steps from Tutorial 1: Basic Energy System.

- Connect the Export to the *ElectricityNetwork.*


Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem
PVPark_e3ec
Aggregated
False
Aggregation Count
1
Angle
0
Asset Type
Commissioning Date
Decommissioning Date
Description
Full Load Hours
0
Id
e3ece454-2c6f-400e-abc4-a
Installation Duration
0.0
Inverter Efficiency
0.0
Name
PVPark_Local
Number Of Panels
Operational Hours
0
Orientation
0
Original Id In Source
Owner
Panel Efficiency
0.0
Power
0.0
Prod Type
Renewable
Short Name
State
Enabled
Surface Area
22287
Technical Lifetime
0.0
Add ports:
Name
Add InPort
Add OutPort







- Assign electricity *Commodity* as a carrier to *Export*.
- Set the power to 10000 Watts, as *Export* has to consume all excess production from the *PVPark*.
- Set the marginal costs of *Export* to 0.01 (a cheap consumer) to make sure that the local electricity demand is met first.

Figure 43 shows the created *EnergySystem* with a *PVPark* and an *Export*.

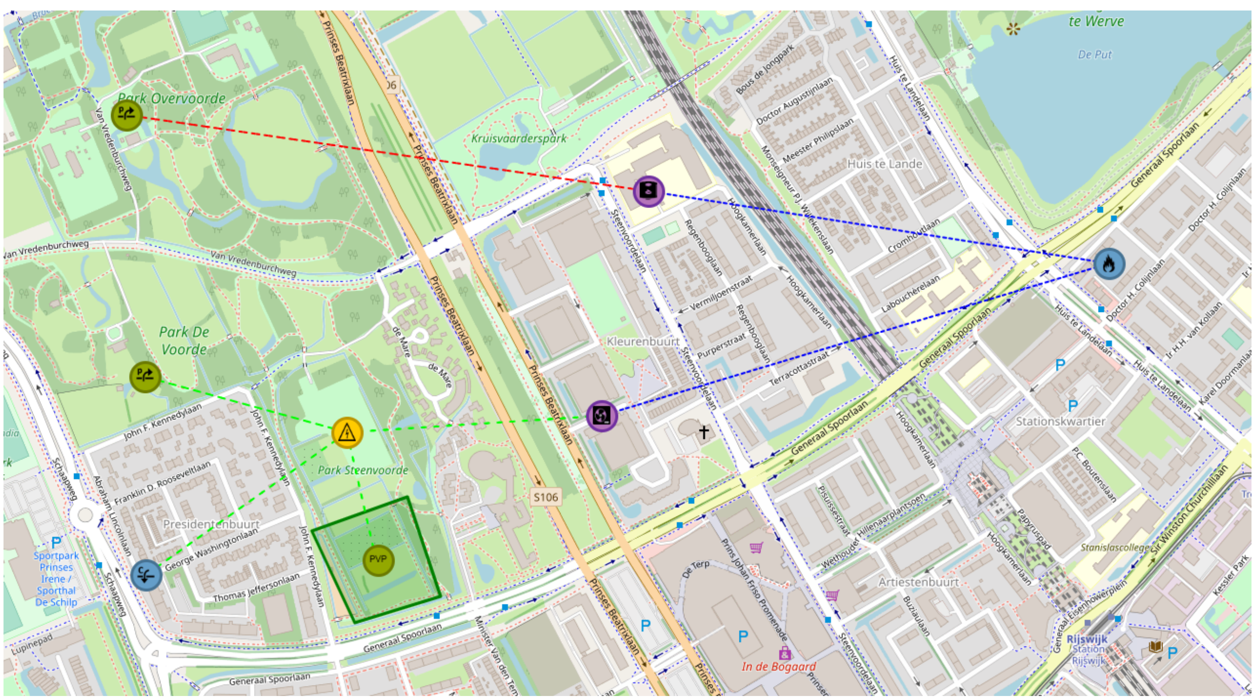


Figure 43: EnergySystem with a PVPark and an Export

Running an ESSIM simulation for this scenario results in balances in all the energy networks. As seen in Figure 44, at times when the *PVPark* generates excess production, it is consumed by the *Export EnergyAsset*, resulting in system balance.

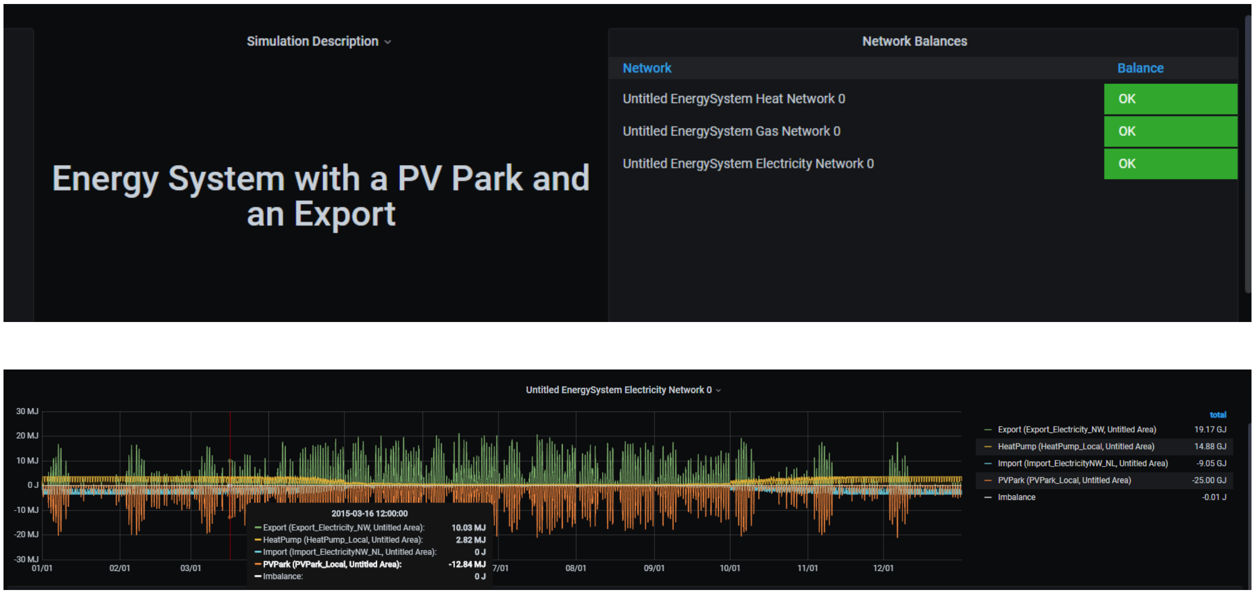


Figure 44: ElectricityNetwork in balance due to Export

7.3.5 Inspecting Load Duration Curves

Sometimes it can be insightful to inspect the loads on the Assets. *MapEditor* offers a quick inspection of the Load Duration Curves (LDCs) of the assets. The LDC displays the hourly values of the load sorted from high to low, thereby showing the frequency of load capacity utilization. For example, LDCs can give an insight into the number of hours energy is imported from and exported to the backbone grid, or for how long energy production or consumption was above a certain threshold. LDCs can be inspected after running an ESSIM simulation.

To inspect an LDC of the *Import EnergyAsset*, follow the steps from Figure 45:

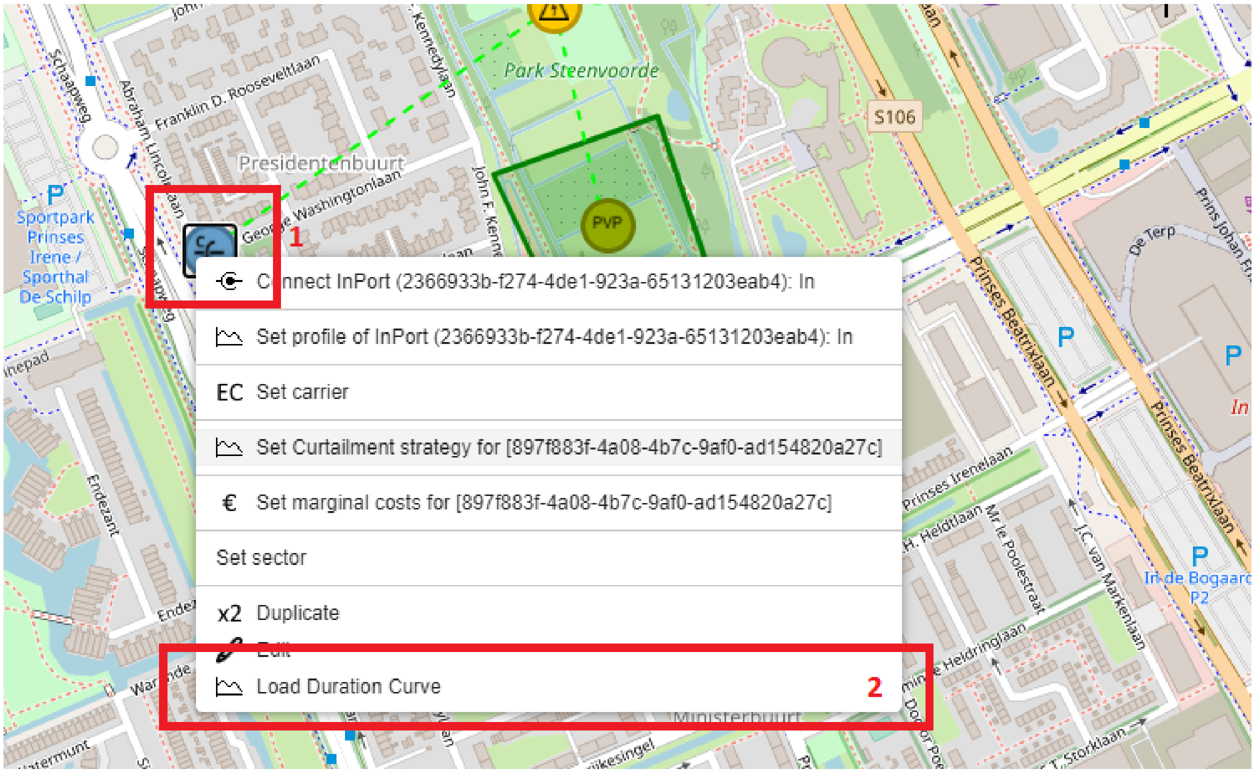
- Right-click on the *Import EnergyAsset* (1)
- Select *Load Duration Curve* (2)

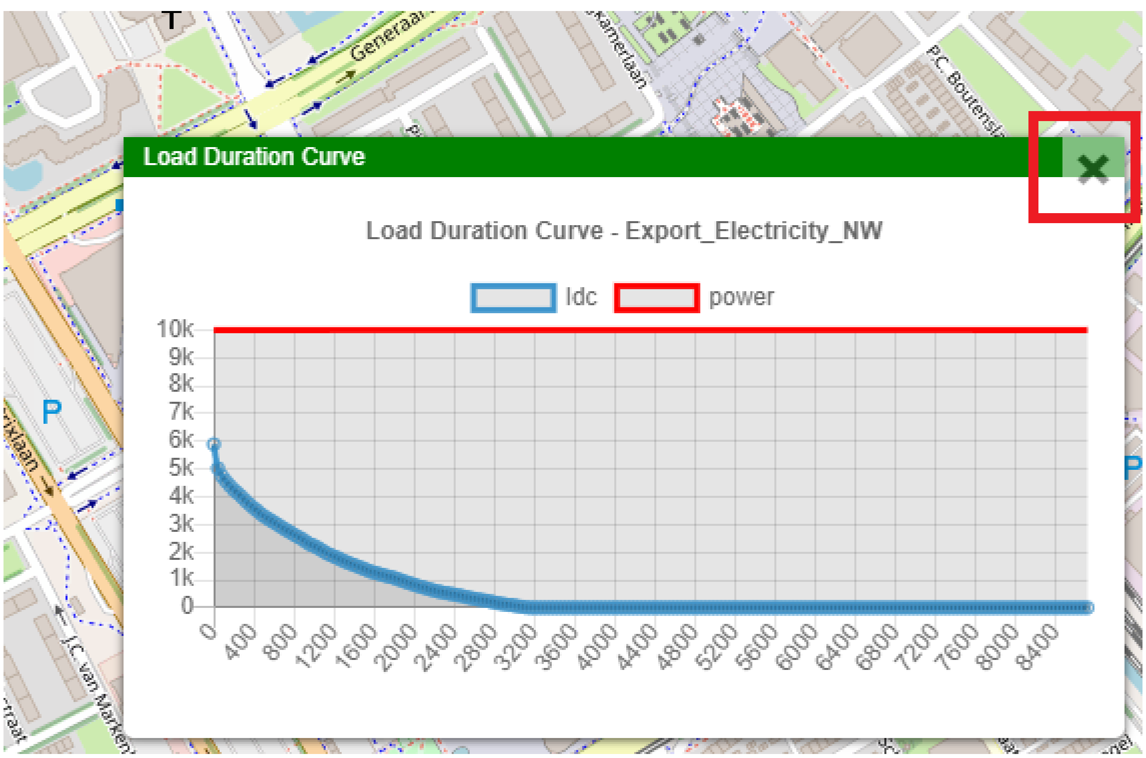
Figure 45: Loading LDC of Export EnergyAsset

A pop-up window opens with Export's LDC (see Figure 46). LDC shows that energy is exported during approximately 3200 hours, and that the peak export for the entire simulation run is around 6k. The maximum power of the *EnergyAsset* is indicated by a red line, and the load stayed well below that.

Figure 46: Load Duration Curve of Export EnergyAsset

- To close the LDC window, select ***x***


Connect InPort (2366933b-f274-4de1-923a-65131203eab4): In
Set profile of InPort (2366933b-f274-4de1-923a-65131203eab4): In
EC Set carrier
Set Curtailment strategy for [897f883f-4a08-4b7c-9af0-ad154820a27c]
€ Set marginal costs for [897f883f-4a08-4b7c-9af0-ad154820a27c]
Set sector
x2 Duplicate
Edit
Load Duration Curve
1
2


Load Duration Curve
Load Duration Curve - Export_Electricity_NW
ldc
power

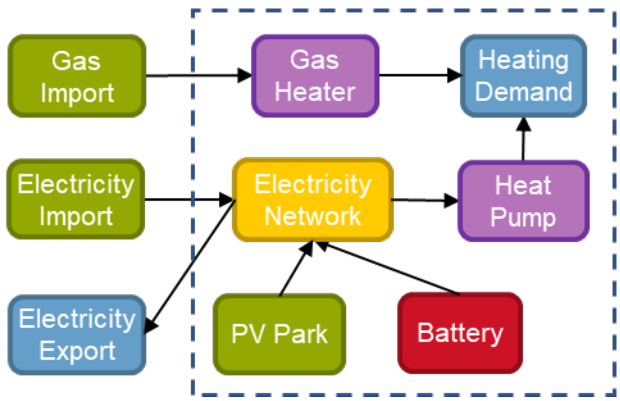
7.3.6 Saving the model

To save the model, follow the steps from Tutorial 1: Basic Energy System, Saving the model. Name the model 'Tutorial3_Scenario.esdl'.

7.4 Tutorial 4: Add storage to prevent export

7.4.1 Description

This tutorial demonstrates an *EnergySystem* that uses local energy storage, a *Battery*, to store overproduction from the local *PVPark*, thereby preventing electricity export. Both the *PVPark* and the *Battery* are connected to a local *ElectricityNetwork*.



7.4.2 Load the model

To build upon the previously created *EnergySystem*, load the 'Tutorial3_Scenario.esdl' file from Tutorial 3: Renewable source export excess. To load the model, follow the steps from Tutorial 2: Not so basic Energy System, Loading the base configuration.

7.4.3 Creating and Configuring a *Battery* Storage *EnergyAsset*

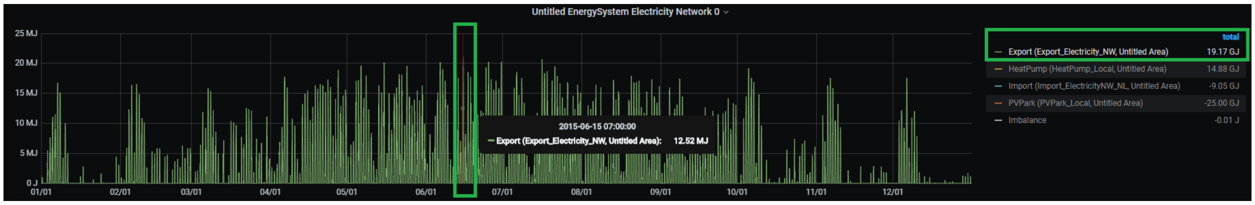
The main aim of this exercise is to prevent export of local *PVPark* production by placing a *Battery*. Configuring a *Battery* storage *EnergyAsset* requires setting a number of parameters such as capacity, charge and discharge rates, and fill level with which the battery starts the simulation. Therefore, to properly configure the *Battery*, we have to first observe the *Export EnergyAsset* from the previous simulation. Run an ESSIM simulation and select the *Export EnergyAsset*.

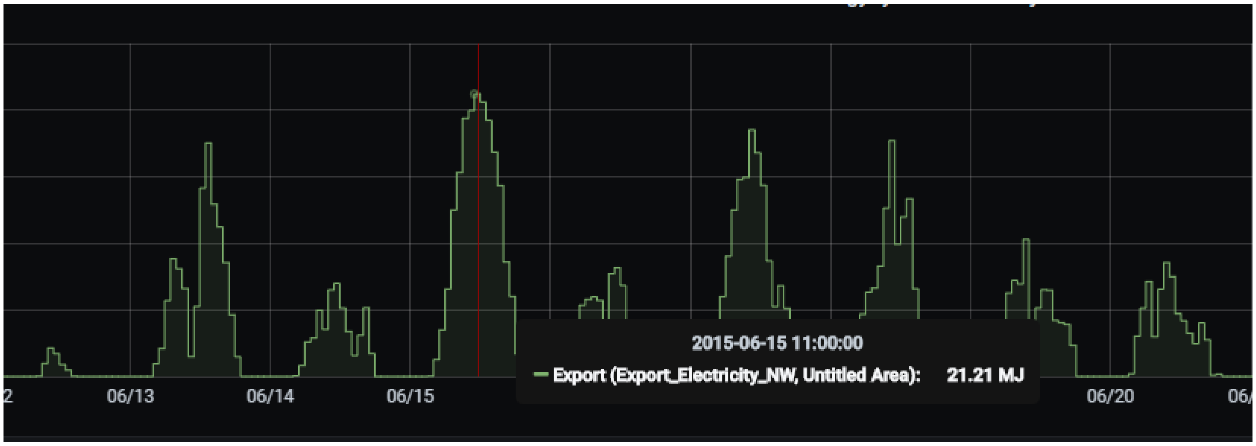
As seen in Figure 47 and Figure 48, the total consumption of Export is 19.17 GJ, and the peak consumption is 21.21 MJ. Configuring a *Battery EnergyAsset* requires setting its capacity, fill level, and maximum charge and discharge rates. Battery capacity indicates the maximum amount of energy a *Battery* can store (in Joules), fill level indicates at what percentage of capacity a Battery is charged at the beginning of the simulation, while maximum charge and discharge rate indicate the maximum power at which a Battery can charge or discharge, at each time step. As the peak demand of export is 21.21 MJ (5891.6 Watts), we can take that value as the maximum charge and discharge rate.

Figure 47: Total and peak consumption of Export EnergyAsset

Figure 48: Peak consumption of Export EnergyAsset

To extend the *EnergySystem* from the previous tutorial, follow the next steps:



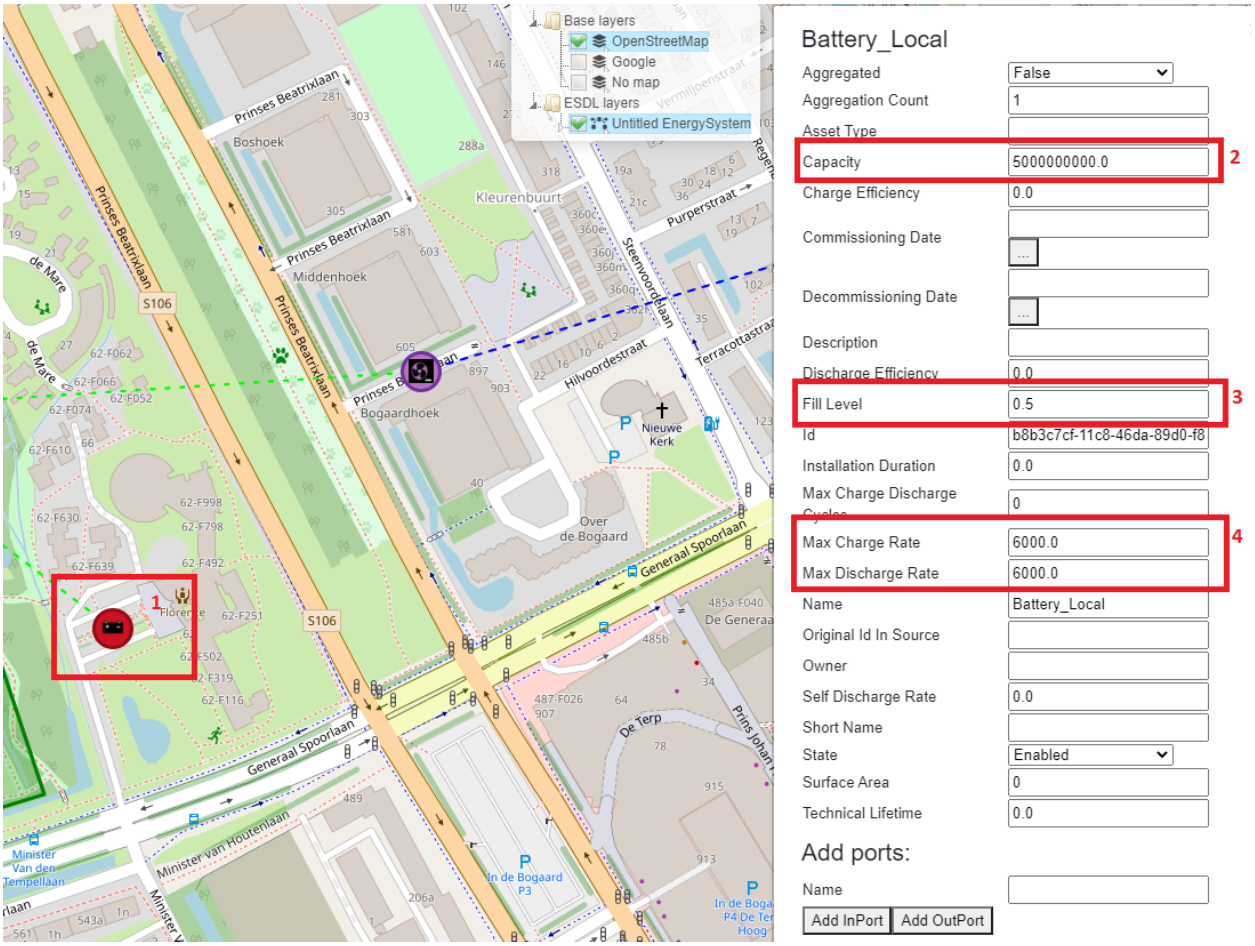


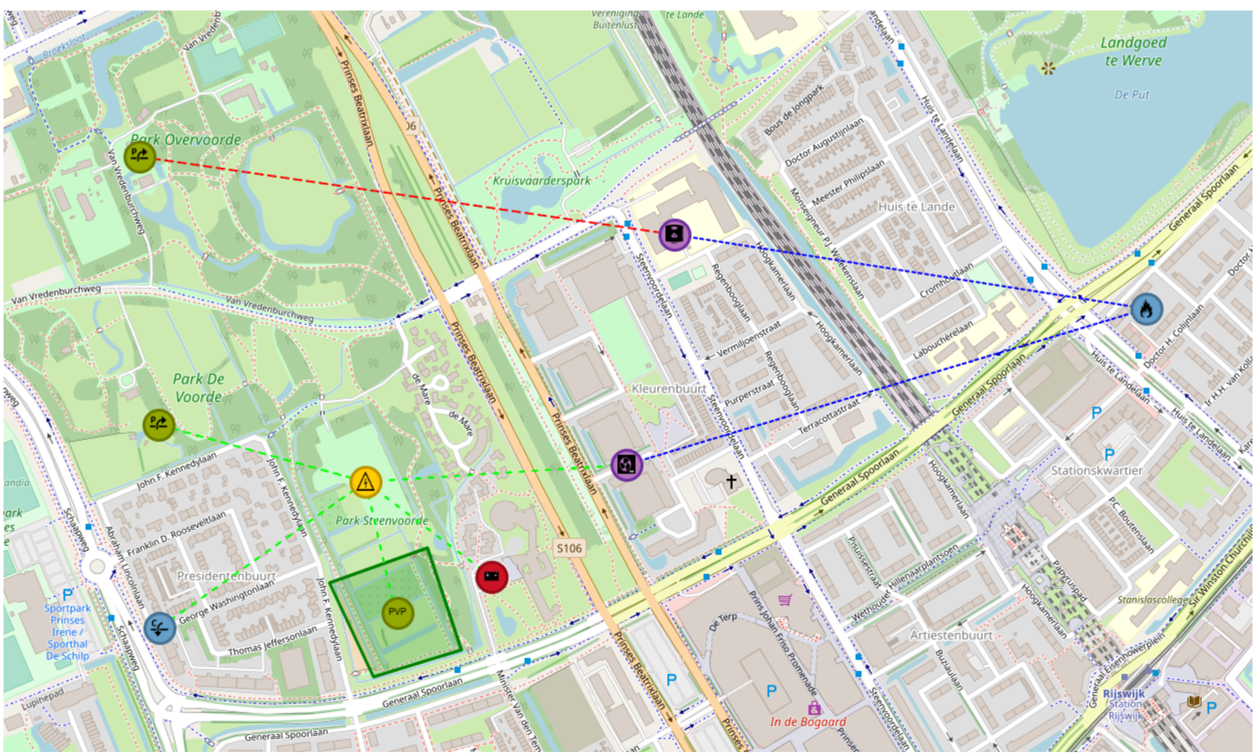
- Create and configure a *Battery EnergyAsset* (see Figure 49)
  - Capacity: 5000000000 Joules
  - Max Charge Rate*:* 6000 Watts
  - Max Discharge Rate: 6000 Watts
  - Fill level: 0.2
  - Set *StorageStrategy* (In earlier versions of this tutorial, the below marginal costs were mixed up)
    - Marginal charge costs: 0.2
    - Marginal discharge costs: 0.8
- Connect the *Battery* to the *ElectricityNetwork*
  - *InPort* of the *Battery* with OutPort of the *ElectricityNetwork*
- Re-assign the electricity *Commodity* to the *ElectricityNetwork*
- Set marginal costs of other *EnergyAssets*
  - *Import*: 0.9
  - *PVPark*: 0.1

Figure 49: Configuring a Battery EnergyAsset

Figure 50 shows the newly configured *EnergySystem*.

Figure 50: EnergySystem with a local Battery storage


Base layers
OpenStreetMap
Google
No map
ESDL layers
Untitled EnergySystem
1
2
3
4
Battery_Local
Aggregated
False
Aggregation Count
1
Asset Type
Capacity
5000000000.0
Charge Efficiency
0.0
Commissioning Date
Decommissioning Date
Description
Discharge Efficiency
0.0
Fill Level
0.5
Id
b8b3c7cf-11c8-46da-89d0-f8
Installation Duration
0.0
Max Charge Discharge Cycles
0
Max Charge Rate
6000.0
Max Discharge Rate
6000.0
Name
Battery_Local
Original Id In Source
Owner
Self Discharge Rate
0.0
Short Name
State
Enabled
Surface Area
0
Technical Lifetime
0.0
Add ports:
Name
Add InPort
Add OutPort
Prinses Beatrixlaan
Generaal Spoorlaan
Minister van Houtenlaan
Kleurenbuurt
Boshoek
Middenhoek
Bogaardhoek
Over de Bogaard
De Terp
In de Bogaard P3
S106


Park Overvoorde
Park De Voorde
Park Steenvoorde
Kruisvaarderspark
Landgoed te Werve
Kleurenbuurt
Presidentenbuurt
Stationskwartier
Artiestenbuurt
Generaal Spoorlaan
Prinses Beatrixlaan
In de Bogaard
PVP
S106

7.4.4 Running and interpreting an ESSIM simulation

To show the effect of placing a *Battery*, run an ESSIM simulation following the instructions from Tutorial 1: Basic Energy System, Running an ESSIM simulation. Figure 49 and Figure 50 show the results for the simulation for *ElectricityNetwork* which show that placing a local *Battery* prevented not only Export, but also Import. The *Battery* starts charged at 20% its full capacity (0.2 fill level). Whenever there is overproduction from the *PVPark*, it is stored in the *Battery*. As *Battery* is configured as a cheaper producer compared to *Import*, it discharges during hours when there is not enough electricity production from the local *PVPark*. Observing the state-of-charge (SOC) of the *Battery*, we can see that it reaches around 55% towards the end of the simulation (see Figure 51).

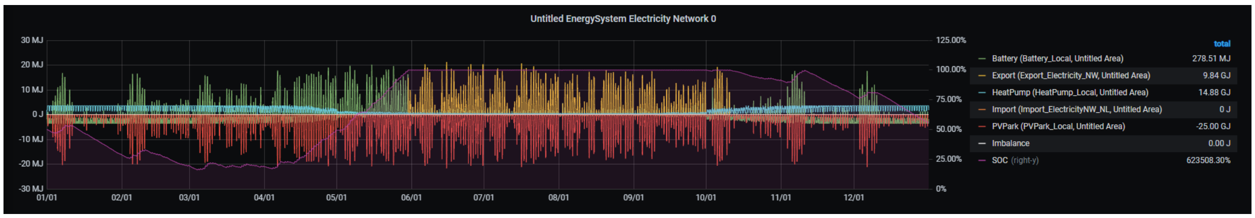


Figure 51: ESSIM simulation results

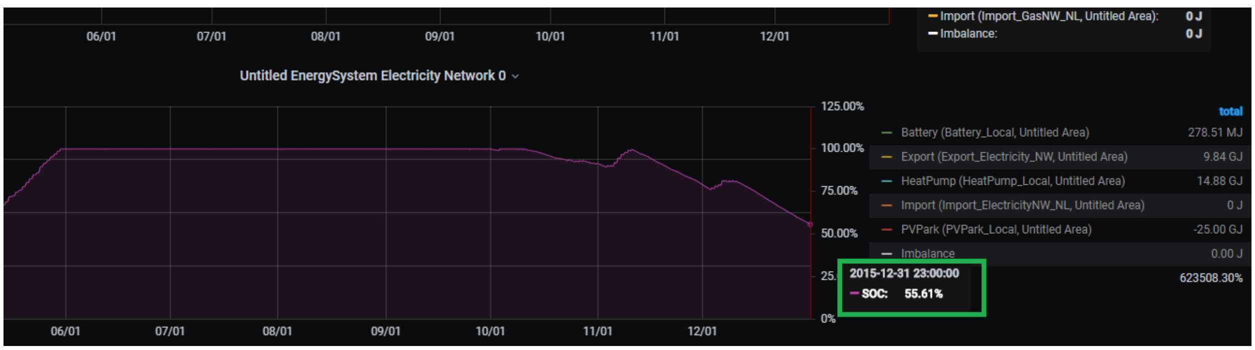


Figure 52: Battery State of Charge

7.4.5 Saving the model

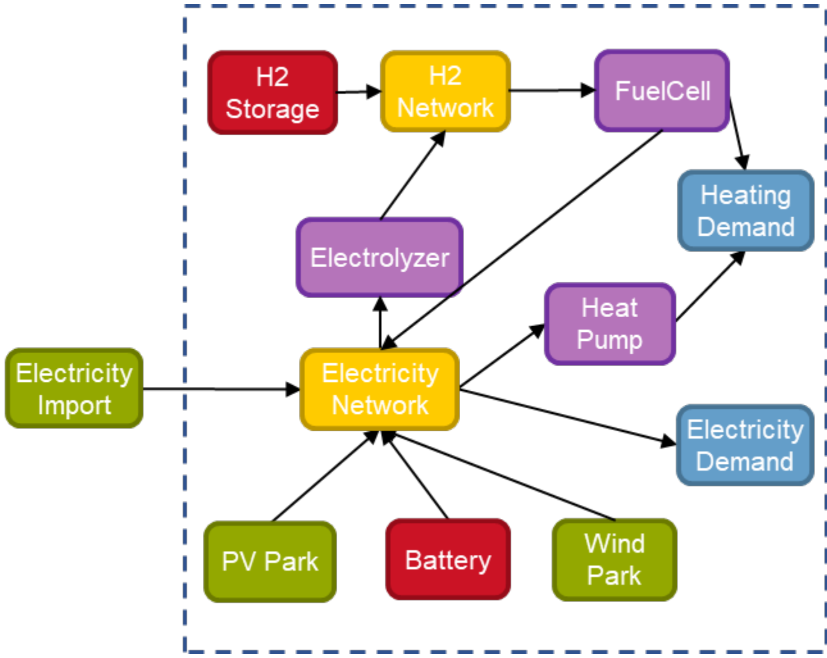
To save the model, follow the steps from Tutorial 1: Basic Energy System, Saving the model. Name the model 'Tutorial4_Scenario.esdl'.

7.5 Tutorial 5: H2 to store excess electricity production

7.5.1 Description

This tutorial guides you through the last and the most complex energy scenario. The scenario demonstrates how excess electricity production from local energy sources is converted t hydrogen (H2) and used in a hydrogen gas network.

In this scenario, a *WindPark* is added as an additional local electricity producer and connected to the *ElectricityNetwork*. Electricity production is used to meet the local *ElectricityDemand* and the demand of the *HeatPump* that partially meets the *HeatingDemand*. Excess electricity production is stored in the *Battery* storage and converted to hydrogen via an *Electrolyzer*. Hydrogen is used by a *FuelCell* to convert it back to electricity and to heat that meets the rest of the *HeatingDemand* (not met by the *HeatPump*). Hydrogen that is not used by the FuelCell is stored in hydrogen storage.



7.5.2 Load the model

To build on the previously created *EnergySystem*, load the 'Tutorial4_Scenario.esdl' file from Tutorial 2: Not so basic Energy System. To load the model, follow the steps from Tutorial 2: Not so basic Energy System, Loading the base configuration.

7.5.3 Creating and Configuring an EnergySystem

In this scenario, the *GasHeater* and gas *Import* are no longer used as heating sources. Therefore, they have to be deleted. To delete these *EnergyAssets*, follow the next steps:

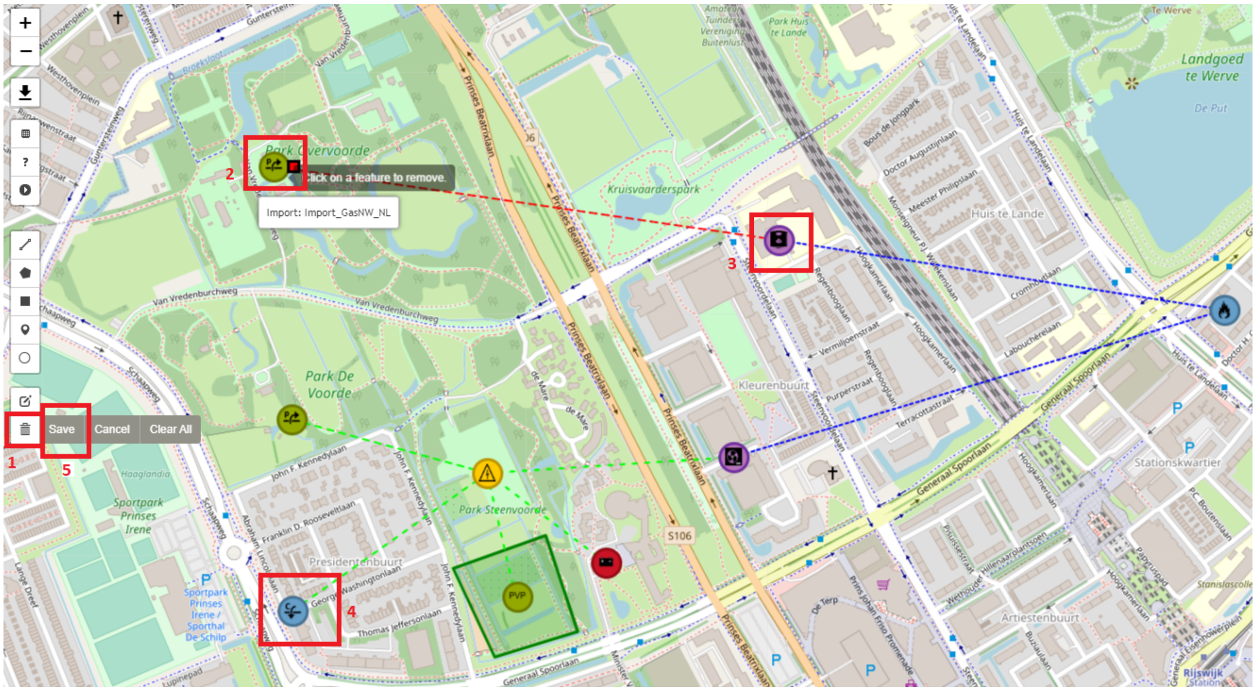
- On the left-hand menu, select the bin icon (see Figure 53) (1)
- Click on the icons of *EnergyAssets* to remove. Once selected, icons will be removed
  - Select gas *Import EnergyAsset* (2)
  - Select *GasHeater EnergyAsset* (3)
  - Select *Export EnergyAsset* (4)
- Click on ***Save*** next to the bin icon to confirm changes (5)
- Refresh the browser to show changes

Gas *Import* and *GasHeater* are now removed.

Figure 53: Removing EnergyAssets

Next, a number of assets have to be added to configure the system for this scenario:

- Create and configure an *ElectricityDemand EnergyAsset.* As *ElectricityDemand* is created and configured in a similar way to *HeatingDemand*, refer to Tutorial 1: Basic Energy System, Creating an EnergySystem for a reference.
  - Name: ElectricityDemand_Local



– Set profile of *InPort*

  * Profile class: *Electricity households (E1A)*
  * Multiplier and Quantity and Unit: 10 GJ

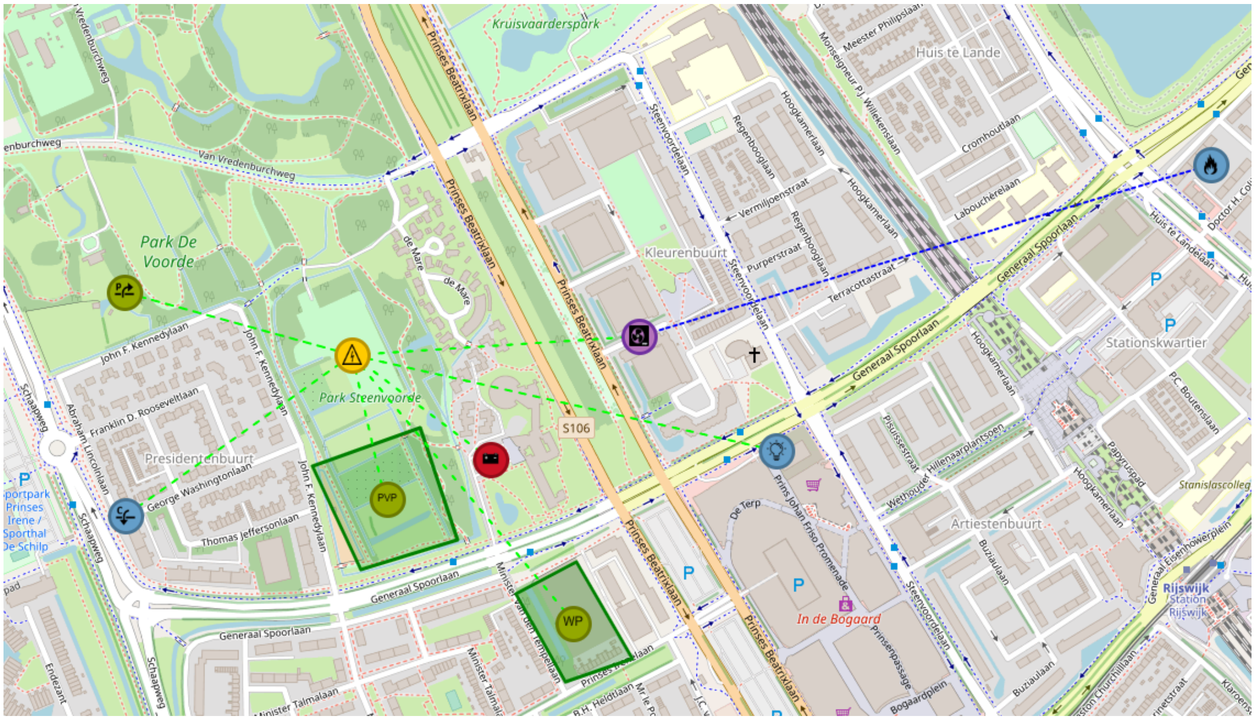
- Create and configure a *WindPark EnergyAsset*. As *WindPark* is created in a similar way to a PVPark, refer to Tutorial 3: Renewable source export excess, Creating and Configuring the ElectricityNetwork, PVPark and Export.
  - Name: WindPark_Local
  - Set profile of *OutPort*
    * Profile class: *Wind op land*
    * Multiplier and Quantity and Unit: 100 GJ
- Connect the *EnergyAssets*
  - *InPort* of *ElectricityDemand* to *OutPort* of *ElectricityNetwork*
  - *OutPort* of *WindPark* to *InPort* of *ElectricityNetwork*
- Re-assign electricity *Commodity* to *ElectricityNetwork*
- Refresh the browser to see the changes

The *ElectricityNetwork* is now configured and looks as in Figure 54.

Figure 54: ElectricityNetwork with an ElectricityDemand and a WindPark

Next, we will create and configure the hydrogen network and its *EnergyAssets*. As MapEditor does not offer an H2Network asset, we can use the *GasNetwork EnergyAsset* to model a hydrogen network, and assign it a hydrogen energy *Carrier*. To create a hydrogen network, follow the next steps:

- Create a *GasNetwork EnergyAsset*. As *GasNetwork* is created and configured in a similar way to an *ElectricityNetwork*, refer to Tutorial 3: Renewable source export excess, Creating and Configuring the ElectricityNet-



work, PVPark and Export for a reference. In the remainder of this tutorial, this network is going to be referred to as *HydrogenNetwork*.

  - Name: Hydrogen Network

- Create a hydrogen energy *Carrier*. Refer to Tutorial 1: Basic Energy System, Creating an EnergySystem on how to add an energy *Carrier*.
  - Carrier type: Energy carrier
  - Name: Hydrogen
  - Energy content: 120000000 MJ/kg
  - State of matter: Gaseous
  - Renewable type: Renewable
- Assign hydrogen energy *Carrier* to *HydrogenNetwork*

An *Electrolyzer EnergyAsset* converts excess electricity from the *ElectricityNetwork* to hydrogen. Therefore:

- Create and configure an *Electrolyzer EnergyAsset* (under *Conversions*)
  - Name: Electrolyzer_Local
  - Efficiency: 0.55
  - Power: 500000 W
- Set *DrivenBySupply* strategy
- Connect *Electrolyzer EnergyAsset* to *HydrogenNetwork*
  - *InPort* of *Electrolyzer* to *OutPort* of *ElectricityNetwork*
  - *OutPort* of *Electrolyzer* to *InPort* of *HydrogenNetwork*
- Re-assign *Carriers* and *Commodities*

  - Re-assign electricity *Commodity* to *ElectricityNetwork*
  - Re-assign hydrogen *Carrier* to *HydrogenNetwork*
- Refresh the browser to see the changes

The newly configured *EnergySystem* should look like Figure 55.

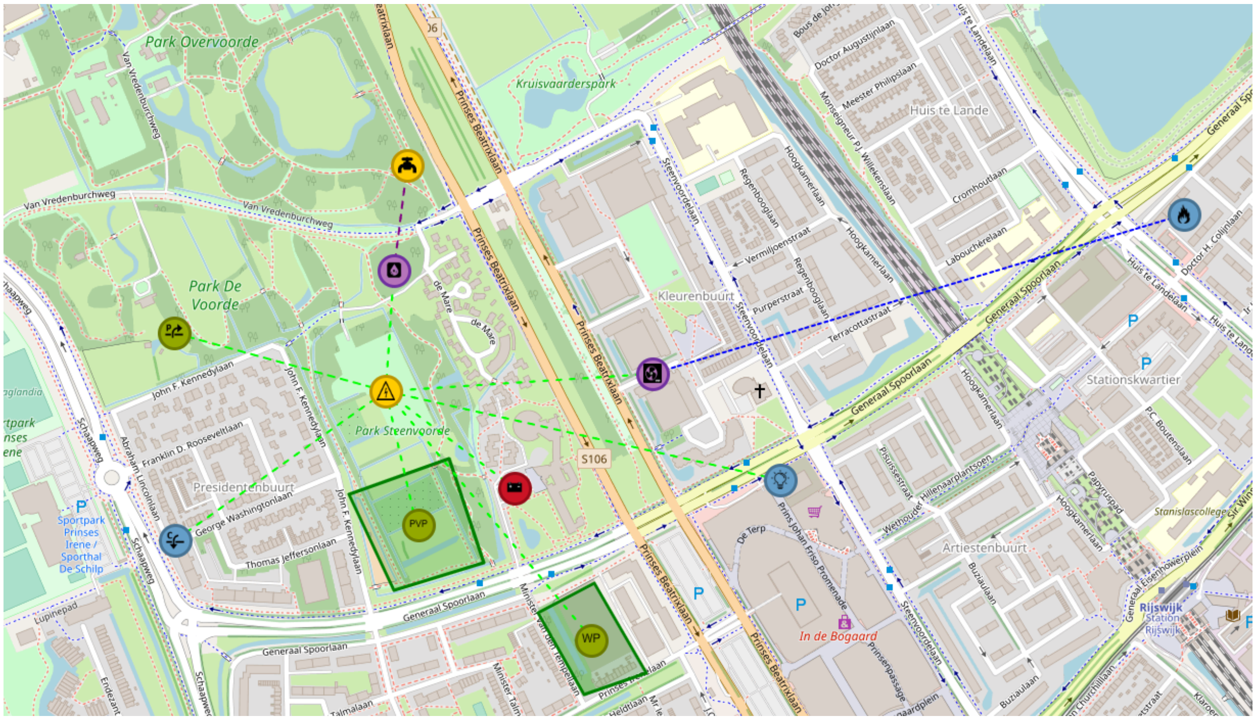


Figure 55: EnergySystem with a HydrogenNetwork and an Electrolyzer

In this scenario, hydrogen produced by the Electrolyzer is used in two ways. Hydrogen is first supplied to a *FuelCell* that converts it back to both electricity and heat. Produced heat is used to meet the rest of the *HeatingDemand* (not met by the HeatPump), whereas electricity is fed back into the *ElectricityNetwork*. Then, if there is excess hydrogen, it is stored in hydrogen *Storage*. To model this scenario, follow the next steps:

- Create and configure a *FuelCell EnergyAsset* (under *Conversions*)
  - Name: FuelCell_Local
  - Efficiency: 0.9
  - Electrical Efficiency: 0.4
  - Fuel Type: Hydrogen
  - Heat Efficiency: 0.6
  - Lead Commodity: Heat
  - Power: 500000 W
- Connect the *FuelCell* to *HeatingDemand*, *HydrogenNetwork* and *ElectricityNetwork*. As it produces both electricity and heat, *FuelCell* has two *OutPorts,* namely **E Out** (electricity) and **H Out** (heat).
  - *InPort* of the *FuelCell* to *OutPort* of *HydrogenNetwork*
  - **H Out** *OutPort* of the *FuelCell* to *InPort* of the *HeatingDemand*
  - **E Out** *OutPort* of the *FuelCell* to *InPort* of the *ElectricityNetwork*

- Set *DrivenByDemand for H Out* strategy
- Re-assign *Carriers* and *Commodities*
  - Re-assign electricity *Commodity* to *ElectricityNetwork*
  - Re-assign hydrogen *Carrier* to *HydrogenNetwork*
- Refresh the browser to see the changes

The newly configured *EnergySystem* can be seen in Figure 56.

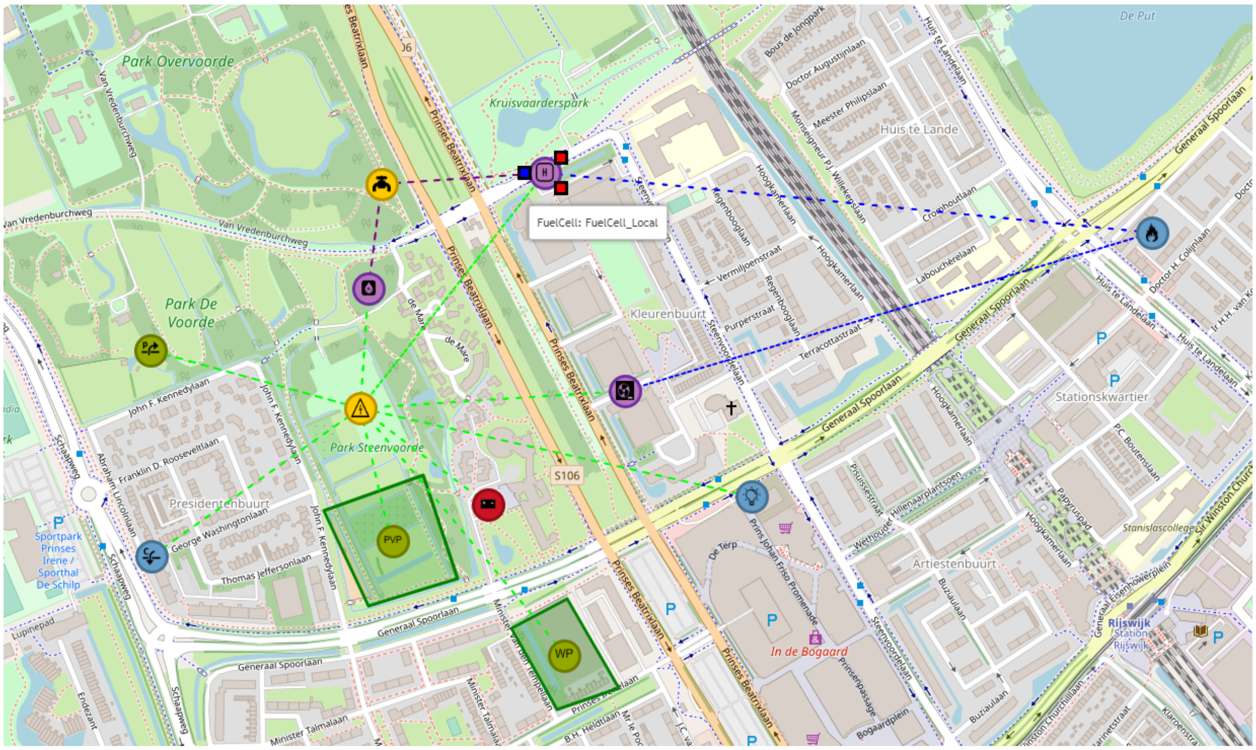


Figure 56: HydrogenNetwork with a FuelCell

7.5.4 Running an ESSIM simulation and interpreting the results

Before creating and configuring a hydrogen *Storage* to store excess hydrogen, first run an ESSIM simulation to see network balances in the newly created *EnergySystem*. This shows the current state of the system, and helps properly dimensioning hydrogen *Storage*. To run an ESSIM simulation, follow the instructions from Tutorial 1: Basic Energy System, Running an ESSIM simulation.

As seen in Figure 57, *HydrogenNetwork* is in imbalance, whereas *Heat* and *ElectricityNetwork* are balanced.

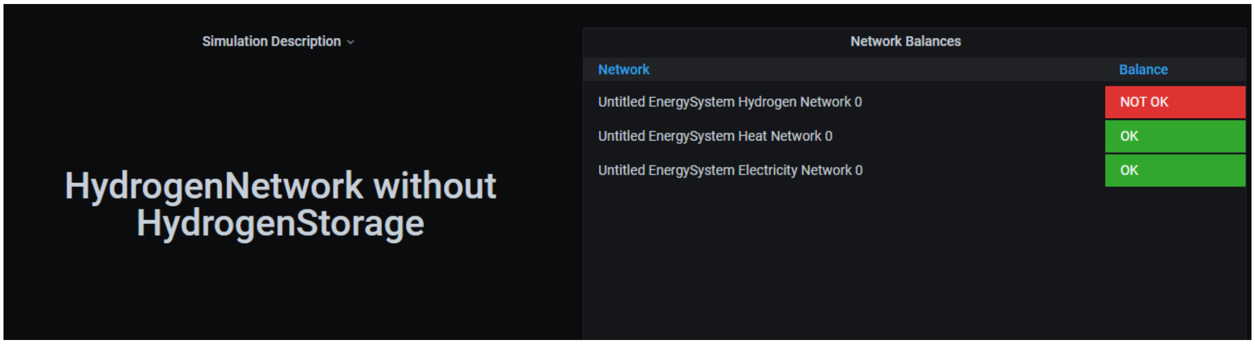
Figure 57: Network balances without HydrogenStorage

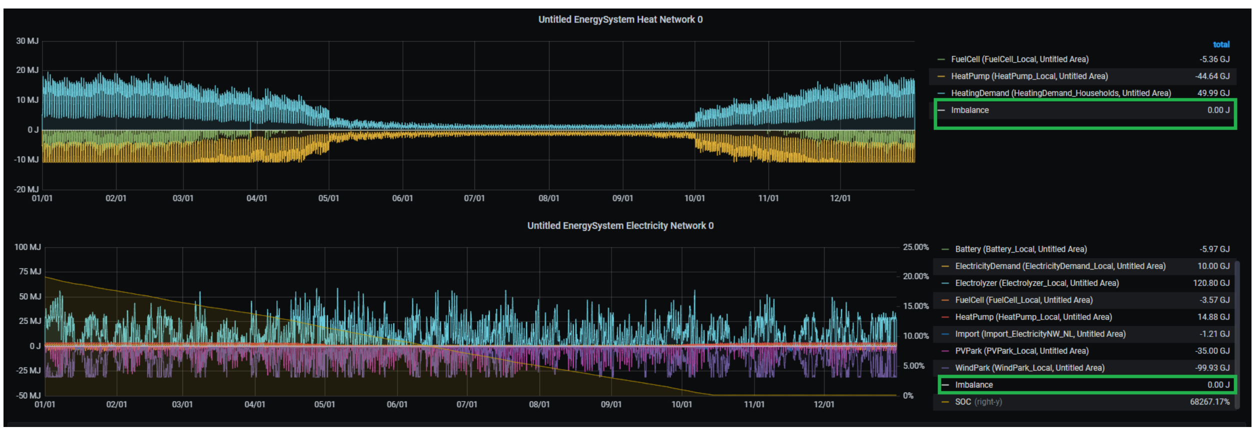
The details can be better observed by looking at individual panels for these networks. Figure 58 shows balance of Heat and Electricity, while Figure 59 shows an imbalance of 57.51 GJ in hydrogen network. A positive imbalance indicates overproduction in the system; therefore, hydrogen *Storage* is needed to store this excess hydrogen.

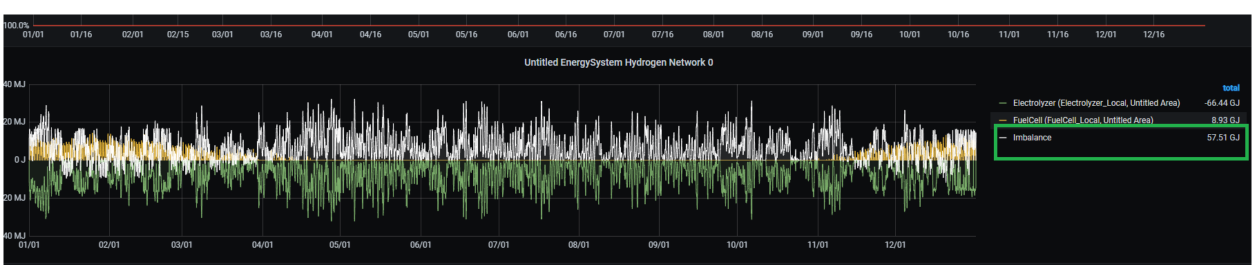
Figure 58: Heat and ElectricityNetwork balances

Figure 59: HydrogenNetwork imbalance

As hydrogen is a gas, hydrogen storage is modelled as a *GasStorage EnergyAsset*. Follow the next steps:


Simulation Description
HydrogenNetwork without HydrogenStorage
Network Balances
Network
Balance
Untitled EnergySystem Hydrogen Network 0
NOT OK
Untitled EnergySystem Heat Network 0
OK
Untitled EnergySystem Electricity Network 0
OK


Untitled EnergySystem Heat Network 0
total
FuelCell (FuelCell_Local, Untitled Area) -5.36 GJ
HeatPump (HeatPump_Local, Untitled Area) -44.64 GJ
HeatingDemand (HeatingDemand_Households, Untitled Area) 49.99 GJ
Imbalance 0.00 J
Untitled EnergySystem Electricity Network 0
Battery (Battery_Local, Untitled Area) -5.97 GJ
ElectricityDemand (ElectricityDemand_Local, Untitled Area) 10.00 GJ
Electrolyzer (Electrolyzer_Local, Untitled Area) 120.80 GJ
FuelCell (FuelCell_Local, Untitled Area) -3.57 GJ
HeatPump (HeatPump_Local, Untitled Area) 14.88 GJ
Import (Import_ElectricityNW_NL, Untitled Area) -1.21 GJ
PVPark (PVPark_Local, Untitled Area) -35.00 GJ
WindPark (WindPark_Local, Untitled Area) -99.93 GJ
Imbalance 0.00 J
SOC (right-y) 68267.17%


Untitled EnergySystem Hydrogen Network 0
total
Electrolyzer (Electrolyzer_Local, Untitled Area) -66.44 GJ
FuelCell (FuelCell_Local, Untitled Area) 8.93 GJ
Imbalance 57.51 GJ

- Create and configure hydrogen *GasStorage EnergyAsset* (under *Storages*). In the remainder of this tutorial, this *Asset* will be referred to as *HydrogenStorage*.
  - Name: HydrogenStorage_Local
  - Capacity: 60000000000 J
  - Max Charge Rate: 10000 W (10 kW to account for the highest peak in imbalance)
  - Max Discharge Rate: 10000 W
- Connect the *HydrogenStorage* to the *HydrogenNetwork*
  - *InPort* of *HydrogenStorage* to *OutPort* of the *HydrogenNetwork*
- Re-assign hydrogen *Carrier* to *HydrogenNetwork*
- Refresh the browser to see the changes

Running an ESSIM simulation with newly created *HydrogenStorage* shows no more imbalance in hydrogen network (see Figure 60). Figure 61 shows that excess hydrogen is now stored in HydrogenStorage.

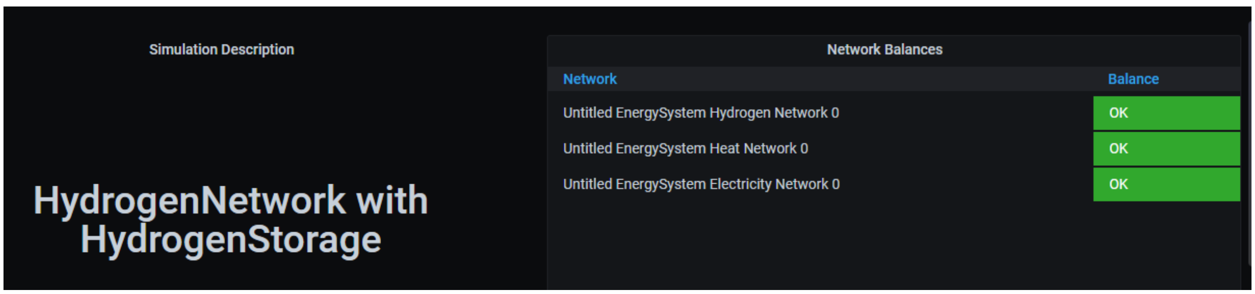


Figure 60: Network balances with HydrogenStorage

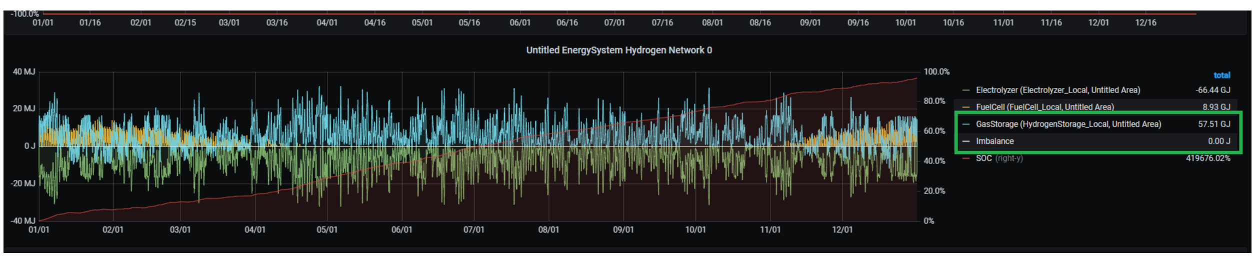


Figure 61: HydrogenNetwork balance with HydrogenStorage

7.5.5 Saving the model

To save the model, follow the steps from Tutorial 1: Basic Energy System, Saving the model. Name the model 'Tutorial5_Scenario.esdl'.

CHAPTER

EIGHT

ESSIM COMMUNITY MEETINGS

This page will be used to share the information that was presented and discussed during the ESSIM community meetings.

8.1 First ESSIM Community Meeting (14th of October 2021)

On October 14th 2021, the first ESSIM community meeting was organized. ESSIM users from different organisations were invited to the event with a focus to discuss the current possibilities, for a look under the hood and to have a sneak peek of what new features and applications lay in the road ahead for the tool. Beforehand, a list of possible topics was sent to the ESSIM users and they were requested to indicate what topic(s) they preferred to have included in the agenda. The preferences of the users were so diverse although one topic (modelling energy flexibility) unanimously stood out from the rest. This made our agenda a lot clearer - we chose to spend one slide on every topic and a few slides on the possibilities of modelling energy flexibility.

To create a similar level of knowledge for all participants and to better understand the explanation of the 'selected topics', we started with presenting the "ESSIM working principles": the topics of control strategies, energy balancing, bid curves and marginal costs and transport solvers were explained.

Then we continued with the selected topics:

- Using real energy market prices in ESSIM simulations
- Support for multi-input multi-output assets
- ESSIM KPI modules
- CO2 calculations
- Using the ESSIM API
- ESSIM combined with loadflow simulations
- Connecting external asset models to ESSIM

This was followed up by a more in depth explanation of the current possibilities of modelling flexibility and the ongoing projects covering this specific topic. This triggered some relevant and interesting questions from the audience, which spurred the subsequent discussion session. With some time to spare, we also took the opportunity to showcase some non ESSIM-related work we're doing with respect to the spatial aspect of the energy transition (including spatial optimization).

The following ESSIM users were invited and most of them were present (in alphabetical order):

- Ecorys
- Ekwadraat (not present)
- ENGIE

- EQUANS (formerly known as ENGIE Services NL)
- MUG Engineering
- Saxion University of Applied Sciences
- Shell (not present)
- Siemens
- Stedin

The reactions from the users were very positive. They found it a very informative meeting and were impressed by the flexibility and possibilities of the ESSIM tool. We agreed to organize a next community meeting in about half a year (March or April 2022).

You can download the presentation (combination of dutch and english slides) `here`

CHAPTER

NINE

INDICES AND TABLES

- genindex
- modindex
- search