

# Improved simulator to analyse the impact of distributed generation on the electricity grid

V. Bakker, A. Molderink, M.G.C. Bosman, J.L. Hurink and G.J.M. Smit  
University of Twente  
Department of EEMCS  
P.O. Box 217, 7500 AE Enschede  
The Netherlands  
Email: v.bakker@utwente.nl

**Abstract**—A change in future electricity grids is expected caused by the introduction of distributed generation, distributed storage and demand side load management. To analyse the impact of these technologies, a simulator has been developed. With this simulator, a small group of households with micro-generators can already be analysed. However, due to the large memory footprint, larger groups of houses cannot be simulated. In this paper an improved simulator which is capable of distributing simulations over multiple PCs via a network is presented.

Using this distributed approach, more (memory) resources can be utilised and more calculations can be done in parallel. Although the introduction of the network stack gives some overhead, still a large speedup is seen when more PCs are used. Furthermore, far bigger groups of houses can be simulated.

## I. INTRODUCTION

Traditionally, most western countries supply domestic electricity demand through generation in large central power stations, with subsequent transmission and distribution through networks. The generation efficiency of the power stations varies between around 35% (older coal stations) to over 50% (modern combined cycle stations), averaging to about 39%. When transmission and distribution losses are considered, the average overall efficiency of the system drops to 35% [1]. This low efficiency is mainly caused by dumping heat produced as byproduct and high fluctuations in demand.

The need to reduce the greenhouse gas effect and increasing energy prices call for efficiency improvements of electricity production, distribution and consumption. This has led to an increase of development of technology that improves the efficiency of electricity production and usage.

Especially generation based on renewable sources like large wind-turbine- and photovoltaic (PV) parks have a high potential since they do not produce carbon dioxide in the production process.

On the domestic level a lot of technologies are also developed. These technologies range from (small) PV installation on rooftops and micro Combined Heat and Power (microCHP) [2] up to controllable appliances [3]. These technologies can be roughly subdivided into three categories:

1) *Distributed Generation (DG)*: Instead of using multiple very large generators, electricity can be produced in smaller, geographically distributed generators. These generators often have a higher efficiency or are based on renewable sources

[1], [2]. Two types of DG can be distinguished, (1) small generation sites on a megawatt level (e.g. Combined Heat and Power, Wind turbine parks) and (2) domestic micro-generators on a kilowatt level [1]. Since these generators are often located closer to locations where the energy is needed, transport costs and losses can be reduced.

2) *Distributed Electricity Storage*: Many houses already have a large heat buffer to optimise the runtime of a in-home heater. Adding large-scale distributed electricity storage, for example in the form of batteries of electrical cars, give an extra option to optimise the production and consumption of electricity.

Inefficiency caused by fluctuating energy demands can be reduced by using these buffers for peak shaving. Furthermore, energy buffers can be used to cope with the fluctuation in production of less manageable renewable sources (e.g. windmills or PV installations).

3) *Demand Side Load Management*: By choosing an opportune moment to switch on/off appliances, thus shifting load, peaks in the electricity demand can be reduced and the generation efficiency can be increased [4]. Even when a appliance has already started, parts of an appliance can be temporarily switched off (e.g. the heating element of a dryer). In extreme cases the appliance can even be switched off temporarily during its runtime (preemption). About 50% of the load in houses is dedicated to appliances that can be switched off or postponed without much discomfort; examples are refrigerators and washing machines [5].

Distributed generation, distributed electricity storage and demand side load management all have the potential to increase the energy efficiency. However, these technologies require a control system. For example, decisions about when to start/stop a generator, charge or discharge a buffer or when to use/not to use a appliance have to be made. Such a control system can have a local and a global scope. Within the local scope a single house is managed. Within a global scope multiple houses in a neighbourhood are managed. Objectives on a local and global scope can be equal, for example peak shaving.

The introduction of (a large fleet of) micro-generators, energy buffers, appliances consuming both heat and electricity

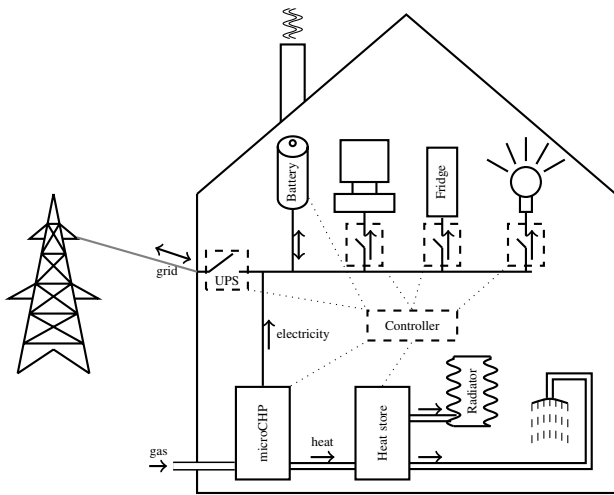


Fig. 1. House schematics

and control algorithms can have a significant impact on the efficiency and reliability of the current generation and transportation infrastructure. To study the effects of the introduction of these elements (devices and controllers) a simulator is developed. With this simulator, already a small group of households with micro-generators can be analysed [6]. However, due to its large memory footprint, larger groups cannot be simulated. In this paper we present an improved simulator by adding networking functionality over for example the Internet. First, in Section II the requirements of the new simulator are described. After presenting the derived simulation model in Section III, details about the implementation and results are given in Section II and Section V respectively. We conclude this paper with the Conclusions and Future work.

## II. REQUIREMENTS

As described in [6], the goal of the first version of the simulator is to create a tool to analyse different combinations of micro-generators, energy buffers, appliances and both local and global control algorithms. This tool should be easily adaptable to new types of micro-generators, controllers and other supported elements. Furthermore, it should be easy to simulate different scenarios and combinations of elements and houses.

The simulated situation should be an accurate model of the actual situation. The modelled house should be able to represent different types of houses and households. Multiple houses grouped together should form a grid, e.g. a city with a realistic mix of houses.

Figure 1 gives an overview of the infrastructure of a single house. Within the house micro-generator(s), energy buffers and appliances are installed. In addition to the current situation all devices within the house (micro-generators, buffers and appliances) are managed (whenever possible) by a local controller. The buffers and micro-generators are observed and managed directly, just like the current situation regarding central heating and the thermostat. The local controller should be able to

query information from the local buffers about the fill level and from the micro-generator about the current generation quantity. Furthermore, it should be able to send start/stop requests to the micro-generator. However, not every micro-generator may be able to follow this request since not every device is controllable (e.g. micro-wind) nor can every micro-generator immediately respond to such a request.

Today, many appliances do not (yet) have a communication interface. Therefore, we assume the appliances are controlled by an overruling switch connected to the controller that can switch off the supply to the appliance. However, the design of the simulator should be such, that it is possible to add more intelligent appliances with a communication interface and management possibilities. All appliances in the house should be separately manageable.

For the simulation of a massive introduction of micro-generation, multiple houses are to be combined into a grid. Each house should be individually addressed because every house has its own characteristics and internal state. The grid can optionally be extended with a global controller communicating with the local controllers, like the situation concerning a Virtual Power Plant (VPP) described in [7]. Due to the different local controllers in the houses, it may be possible that not all houses respond to the global controller because not every type of local controller reacts on the signals in the same way (which is a realistic scenario).

The grid has to keep track of the total import/export of electricity within the grid. This data can be used to determine the success of the controllers. Furthermore, the grid should be capable of limiting the maximum amount of electricity that can be imported from or exported to the grid.

Since the purpose of the simulator is to simulate various new scenarios and elements, the simulator has to be flexible. Adding new types of micro-generators, buffers and appliances has to be easy, just as changing control algorithms. Some elements act upon both heat and electricity, for example the earlier mentioned hot-fill washing machine and microCHP devices. Therefore, for all devices the behaviour concerning both heat and electricity has to be defined. The combination of heat and electricity is referred to by "the energy profile" in the rest of this paper.

Since it is a discrete simulation (due to modelling choices), the length of the time intervals is the choice which mainly influences the amount of data. It is a tradeoff between precision and data usage. The minimum possible time interval length may be a second, but this has to be adjustable. A five minute time interval is a good tradeoff between precision and data usage [8]. The precision of the data itself does not form a major issue, since almost all data can be stored as integers representing Watts.

The last requirements concern the speed and memory usage of the simulator. In the initial simulator, around 50.000 households can be simulated using very simple control algorithms. After this amount, the simulator is limited by the memory available in the computer. While the control algorithms have evolved, so have their memory requirements. Furthermore,

we want to simulate a larger group than 50.000 households. To achieve this, the simulation can be split up in smaller parts and distributed over multiple computers via a network. However, the network overhead should be limited. In the best case, the increase in computational power by spreading the simulation over multiple computers will compensate for the network overhead.

Since multiple computers are used for a simulation, information required for simulation is needed on multiple places. Thus, the simulator must be able to distribute all the required information, without human interaction. When some parameter has changed, these changes should be propagated automatically.

### III. MODEL

Based on the house schematics in Figure 1 and the (new) requirements given in the previous section, the (adjusted) model is depicted in Figure 2.

The model of the house has not changed. Every house consists of (several) micro-generators, heat and electricity buffers, appliances and a local controller. Micro-generators can produce heat and electricity. All available micro-generators are modelled in this way, considering that the generation can be zero or even negative. All appliances in the house are modelled as combined electricity and heat consumers (where consumption of an energy source can be zero). This way both heat as electricity consuming appliances can be modelled, as well as appliances that consume both (for example a hot fill washing machine). All consumers are defined as appliances, from a fridge and a coffeemaker to central heating and tap water. Buffers and heatstores are temporary electricity and heat storages.

Multiple houses are combined in a grid, exchanging electricity and information. However, the model of the grid has changed to “distribute” the houses over the network. As depicted in the figure, the houses are distributed over multiple so called subgrids. The houses, these subgrids behave like the initial grid model. Each client has a subgrid, where the subgrid communicates with the global grid to exchange the required information.

Within the model the planning horizon is discretized resulting in a set of consecutive time intervals. The number of intervals depends on the length of the planning horizon and the length of the intervals. In general we use an interval length of six minutes and a planning horizon of one day, resulting in 240 time intervals.

### IV. IMPLEMENTATION

A simulation starts by, based on the simulation configuration, initialising a simulation. During the construction of the simulation, a grid and optionally a grid controller is constructed. During the construction of the grid, the houses connected to this grid are constructed. During the construction of the houses, all the appliances, buffers, generators etc. are constructed. When all required entities are constructed, all the time intervals can be simulated. For each time interval, the grid is ‘ticked’. The grids ticks the houses, the houses

the appliances, buffers, generators etc. During such a tick, each entity updates its internal state and energy/information between the entities is exchanged.

In the networking simulator, the households are distributed over the network, which requires a change in the grid. The houses that need to be simulated are divided over the connection clients automatically. Each clients has its own subgrid to which the houses are connected (see Figure 2). A global grid, located at the server, controls all subgrids via the network connections. At the end of the simulation, data of all the subgrids is aggregated in the global grid.

#### A. Protocol

One of the requirements of the improved simulator was the ability to simulate larger groups of houses. By distributing the houses over multiple PCs, the memory and computational requirements of the simulation are spread over multiple PCs. However, the network overhead should be limited. Thus a fast, highly efficient protocol is required.

Since a simulation is distributed over multiple PCs, all information required for a simulation must be available at each client. The protocol must thus be able to distribute all required information prior to the construction of the (sub)grids, households etc.

Based on the flow of a simulation, a protocol between the server and the client is developed (see Figure 3). After the client has connected to the server, a welcome message is sent to confirm a simulation client is connected and not some random process.

When the user selects which simulation configuration has to be simulated, the server automatically distributes the required houses over the connected clients. Simulation configurations are simple files, which can be easily transferred. For each client, a new configuration file of the subgrid is created, and information about this file is sent via a `SimulationMessage`. When the client retrieves this `SimulationMessage`, it first looks if it already has a cached version of the configuration file.

If it has a cached version, it has to check whether this file has not changed since it was obtained. This is done by sending a `ConfigInfoMessage`, which contains the filename of the configuration and a hash of it’s contents. When the server retrieves a `ConfigInfoMessage`, it constructs a `ConfigInfoMessage` from it’s own configuration and sends it back to the client. When the hashes are equal, the file is up-to-date.

If the hashes are not equal, or when the configuration file did not exist at the client, a `RequestConfigMessage` is sent. When the server receives such a message, the file is read from disk, compressed and sent to the client.

When the configuration file is up-to-date at the client, it is scanned for dependencies. For example, a grid consists of houses, houses have their own configuration (files). For each dependency, their configurations files are exchanged in a similar way.

When all configuration files are up-to-date, the client confirms it has finished the configuration phase by sending a `SimulationMessage` to the server.

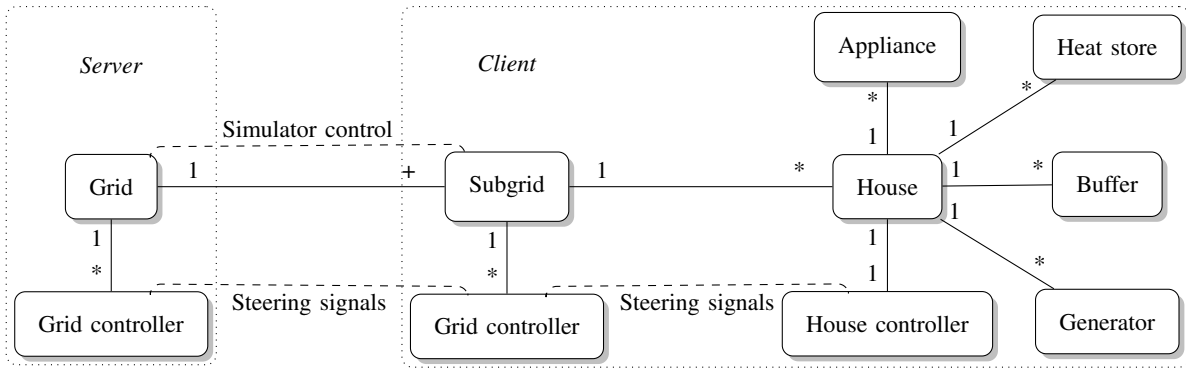


Fig. 2. The model

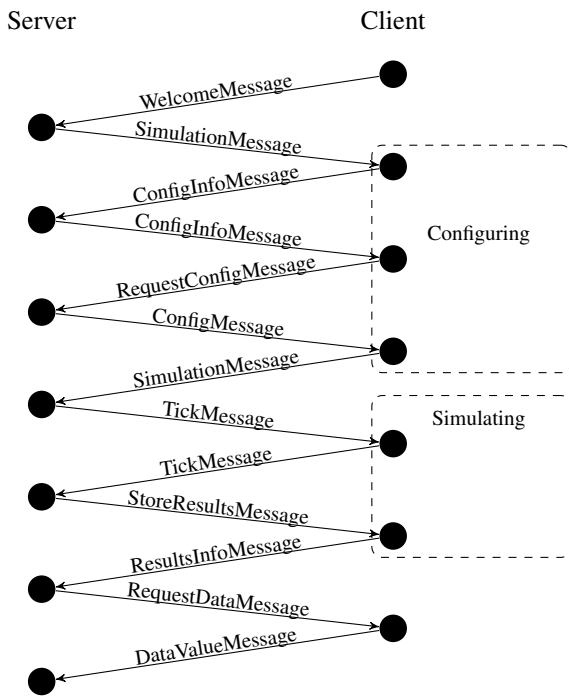


Fig. 3. The used protocol

When all clients are configured, the server sends a TickMessage to simulate the first tick. Each client simulates the time interval and confirms this interval to the server when they are done by replying with a TickMessage. The server waits for all client to confirm their tick, and sends a new TickMessage to each client. This way, all the clients stay in sync with each other, which can be required when simulating some global optimisation algorithm.

When all time intervals are ticked, a StoreResultsMessage is sent. During this phase, statistics and information about the whole simulation is calculated at the clients. The completion of this phase are confirmed by each client with a ResultsInfoMessage. This ResultsInfoMessage contains information about which simulation results are available at the client. Note that

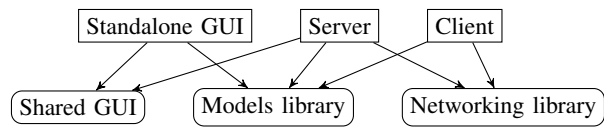


Fig. 4. The software architecture

all the results are still stored distributed over the clients, the server only collects where the information is stored.

Since the grid is split up into subgrids, the server combines the subgrids into a global grid by aggregating the data. When data of a house or grid is required, the server request the data by sending a RequestDataMessage to the client which has this data. The client sends the required information to the server with a DataValueMessage. After requesting and aggregating all grid data into a global grid, the GUI on the server is informed that the simulation is complete.

When the simulation has to be saved to disk, or when information about the simulation has to be displayed in the GUI, the server request the required data from the clients on demand. This way, the amount of memory required on the server and the amount of data transferred over the network is limited.

By disconnecting the client, the simulation results are discarded at the client. The client automatically reconnects to the server, and the process starts again from the beginning.

### B. Software architecture

Based on the model and the developed protocol, the simulation is split up into multiple parts (see Figure 4). The basis of the simulator (the presented model) has been rewritten into a library. Since both the Server and the stand-alone GUI share user interface elements to configure all the entities in the model, a library to configure the simulations has been written.

A 'stand-alone' simulation without a server/client model (the simulator we started with) can be used for smaller simulation instances.

Newly added to the code base was the simulation protocol and the required C++ classes for a server and the client. The server is similar to the stand-alone software. With this

Used simulator	1 house	5.000 houses	10.000 houses
Standalone	0.2s	375s	-
1 Client	0.6s	416s	-
2 Clients	-	311s	720s
3 Clients	-	211s	475s

TABLE I  
EXECUTION TIMES OF SIMULATIONS

program, the entities can be configured and simulation configurations can be created. When one or more clients are connected to the server, simulations can be performed and their results can be displayed. Note that the server cannot simulate a configuration, this always has to be delegated to a client.

The client is responsible for the real simulation. Based on the configuration received from the server, the real model is calculated at the client.

## V. RESULTS

The software is developed using the cross-platform QT library [9] in C++. Versions for Linux, Microsoft Windows and Mac OS X have been built and deployed. To analyse the performance and impact of the network library, three simulation scenarios have been simulated. In each simulation, our the state of the art house controller is used. This controller has very high computational and memory demands. Although 5.000 houses is still possible with a modern PC, 10.0000 houses with this controller is already infeasible.

The simulations are performed using the stand-alone simulator and with the server/client simulator using one up to three clients. In all the networked simulations, the machine running the server also ran a client.

In the first scenario, only a single house is simulated. In the second scenario a grid contains 5.000 houses, in the third scenario 10.000 houses. As stated above, it already is infeasible to simulate 10.000 houses using this house controller on a single modern PC (2GHz processor, 2 GB RAM). The execution times of these simulations are depicted in Table I.

When using the network simulator with only one local client, the connection is made via localhost. The localhost interface is a very fast interface, so no effects of network congestion or latencies have occurred.

As can be seen in the table, there is a overhead added by the networked simulation. This is caused by the distribution of the configurations over the network interface and the aggregation of the subgrids.

Looking at the first scenario, there is a significant overhead compared to the standalone simulator. However, the network overhead is still below one second, which is acceptable.

What is interesting to see, is that in the second scenario there is a very big difference in execution times compared to the standalone simulator. Although the amount of configuration data and the amount of information about the subgrid is the same, the aggregation of the (single) subgrid into the grid is a very expensive operation (taking roughly 40 seconds). This is caused by the increased amount of information that has

to be transferred to the server about which simulation results are available at the client(s). This also explains the decrease in execution times when more clients are used. The amount of generated simulation data is roughly the same (only one subgrid is added per client), so the speedup is caused by adding more computational power to the simulation.

As mentioned above, simulating 10.000 houses was already infeasible using a single computer. With the network simulator, this becomes possible. When two clients are used, 5.000 houses are simulated per client. Considering the execution time of the standalone simulator (375 seconds) and the execution time of the simulation of 10.000 houses using two clients (720 seconds), the overhead caused by the network simulation is acceptable compared to the total simulation time since there still is a speed-up.

## VI. CONCLUSION & FUTURE WORK

In this paper, a new improved simulator has been presented. The result of the work is a new, modular and fast simulator capable of simulating large groups of households by distributing the work over multiple PCs via a network. The amount of user interaction is minimised by building a fully autonomous client which requires no set-up. All simulation configurations are distributed and synced over the network, so for the end user hardly any change is notable.

Although the new simulator works well, still some improvements are possible. Currently, still only one processor (thread) is used while simulating a (sub)grid. Since all the houses can be simulated individually, multiple threads might be used to spread the workload over multiple processors. Furthermore, the aggregation of the grid is an expensive operation. Perhaps it is possible to aggregate some simulations after each tick (in parallel to the next ticks) to speed up this process.

## ACKNOWLEDGMENT

This work is sponsored by Essent and GasTerra.

## REFERENCES

- [1] A. de Jong, E. J. Bakker, J. Dam, and H. van Wolveren, "Technisch energie- en CO<sub>2</sub>-besparingspotentieel van micro-WKK in Nederland (2010-2030)," Werkgroep Decentraal, Tech. Rep., July 2006.
- [2] United States Department of Energy, "The micro-CHP technologies roadmap," *Results of the Micro-CHP Technologies Roadmap Workshop*, December 2003.
- [3] D. Hammerstrom, R. Ambrosio, T. Carlon, J. DeSteele, G. Horst, and R. Kajfasz, "Pacific northwest gridwise testbed demonstration projects, part i and ii," Pacific Northwest National Laboratory, July 2007.
- [4] A. Peacock and M. Newborough, "Controlling micro-chp systems to modulate electrical load profiles," *Energy*, vol. 32, no. 7, pp. 1093–1103, July 2007.
- [5] C. Block, D. Neumann, and C. Weinhardt, "A market mechanism for energy allocation in micro-chp grids," in *41st Hawaii International Conference on System Sciences*, Jan 2008, pp. 172–180.
- [6] A. Molderink, M. Bosman, V. Bakker, J. Hurink, and G. Smit, "Simulating the effect on the energy efficiency of smart grid technologies (*accepted*)," in *Winter Simulation Conference 2009*. IEEE, 2009.
- [7] J. Scott, P. Vaessen, and F. Verheij, "Reflections on smart grids for the future," Dutch Ministry of Economic Affairs, Apr 2008.
- [8] A. Wright and S. Firth, "The nature of domestic electricity-loads and effects of time averaging on statistics and on-site generation calculations," *Applied Energy*, vol. 84, no. 4, pp. 389–403, April 2007.
- [9] Nokia. Qt software library. [Online]. Available: <http://qt.nokia.com>