A Self-Learning Energy Management System for a Smart-Grid-Ready Residential Building

Fabiano Pallonetto
Electricity Research Centre
University College Dublin
Dublin, Ireland
fabiano.pallonetto@ucdconnect.ie

Yerlan Turenshenko
School of Computer Science
University College of Dublin
Dublin, Ireland
yerlan.turekeshov@ucdconnect.ie

Eleni Mangina
School of Computer Science
University College of Dublin
Dublin, Ireland
eleni.mangina@ucd.ie

Donal P. Finn
School of Mechanical and Materials Engineering
University College Dublin
Dublin, Ireland
donal.finn@ucd.ie

ABSTRACT

Based on research and scientific advances in sensor and network technologies, machine learning, and standard statistical methods, a development and a deployment of energy management systems could reduce the cost of electricity in residential buildings. This paper describes two implementations of an energy management system. The objective of the algorithm is to reduce the energy consumption of a residential building maintaining the thermal comfort. The first prototype used a rule-based control flow and reduced the baseline consumption by 25%, whereas the smart version of energy management system reached almost 50% minimisation of consumption by predicting future changes in the house temperature via a tree based machine learning models generated in R language. This Smart Controller with these predictions and energy cost calculations makes decision to either turn on or off the heating system of the house. To test and evaluate the system, both energy management systems run a virtual building simulation environment such as EnergyPlus through its interface controller BCVTB and RESTful API service that controls the building simulation software and stores obtained results to its database.

Keywords

energy management system, heating system, energy consumption, smart controller, building simulation, optimisation, residential building

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*; G.1.2 [**Approximation**]: Minimax approximation and algorithms; I.6.3 [**Simulation and Modelling**]: Applications

1. INTRODUCTION

In a power system, increasing levels of renewable energy sources results in additional system variability that could cause blackouts and technical unbalances between supply and demand. Latest forecast techniques and flexible generation can mitigate such challenges, as well as flexible demand resources [3]. The latter measure is being implemented worldwide through the remuneration of demand response aggregators in the electricity market. Such evolution in the regulation is one of the main drivers for the wide interest in developing Energy Management System (EMS) that could implement demand response strategies in residential buildings [1]. Buildings of the future equipped with smart EMS and new electric heating systems could have the capabilities to be responsive to smart grid signals. Such feature enables buildings to reduce or increase their electricity consumption as leaf element of the power system [6]. Demand response in residential buildings could economically and technically be viable if dwellings will have a smart control of the electric loads responsive to demand response signal and weather forecast. Such smart controller could be embedded in a Home Area Network (HAN) and become part of the dwelling system. The effectiveness of smart controls inside residential buildings in terms of energy saving and flexibility supplied to the power grid could be evaluated through the utilization of building simulation software. Building simulation models can be save considerable resources compared to experimental analysis and are generally accurate to within a 10% range. Furthermore, building simulation software could be interfaced to sophisticate control system and the results could be evaluate across different years without a hardware and communication network prototypes. This work aims to develop to design and develop an EMS and subsequently to test its performance

SIMUTOOLS 2015, August 24-26, Athens, Greece

DOI 10.4108/eai.24-8-2015.2261068

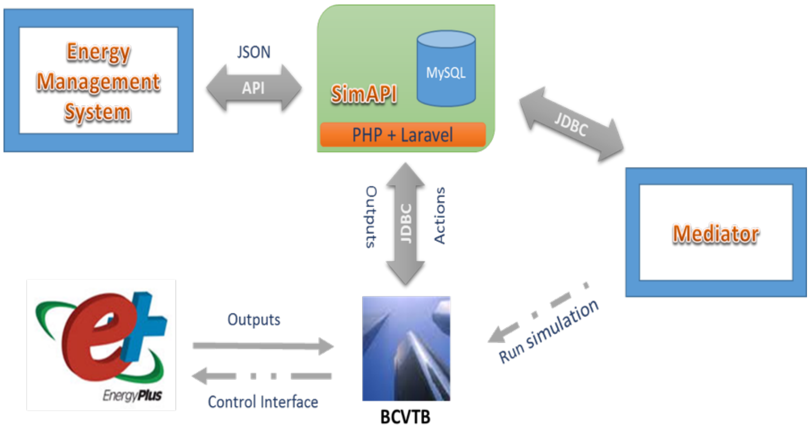

Figure 1: Software architecture of the testing environment

Table 1: Electricity tariff

Time	Weekdays	Holidays and weekends
	€/kWh	€/kWh
23:00 - 08:00	0.09	0.09
08:00 - 17:00	0.125	0.125
17:00 - 19:00	0.38	0.125
19:00 - 23:00	0.125	0.125

Table 2: Building thermostatic set points

	Weekdays	Weekend
00:00 - 06:30	19	20
06:30 - 09:00	19	20
09:00 - 16:00	16	20
16:00 - 00:00	18	20

using a EnergyPlus building model of a dwelling located in Ireland.
The paper is organised as follows: Section 2 describes the overall architecture of the general system while Section 3 analyse the key components of the first and second designed energy management system. The results and evaluation stages are covered in the Section 4. In addition, this section will briefly discuss the difficulties encountered in the testing of the system, give a detailed account of the approach taken in addressing these issues, and, in each case, justifies the reasons for choosing a particular approach. Finally, Section 5 concludes the paper and proposes possible future enhancements to this energy management system.

2. SOFTWARE ARCHITECTURE

The software architecture of the system used to benchmark and assess the performances of the EMS is illustrated in Figure 1. The developed energy management system is connected with the building simulation model using an open source software called SimApi [8]. This software implements a restful API to allow a remote control of a building simulation. The API is developed in PHP 5.5 and using Laravel 4.2 as MVC framework. At each simulation time step, SimApi stores all the API requests in a MySQL relational database [7] while it retrieves the sensors readings from the building simulation. The communication between the database and the building simulation software is developed using the Building Control Virtual Test Bed (BCVTB) [14]. BCVTB and the SimAPI are connected to the database via Java DataBase Connectivity (JDBC) technology. The EMS at each timestep sends through the API a JSON with the thermostatic settings of the building and the state of the heating system (ON/OFF) for the next timestep. Such JSON with the thermostatic set points, and the state of the heating system for the next timestep is called action. For each simulation, the EMS sends n-1 actions that represent the control flow of the system. A Java package called mediator acts as simulations manager. When the EMS invokes a begin simulation end point in the API, the mediator transfers the appropriate EnergyPlus file and run the BCVTB and EnergyPlus.

3. ENERGY MANAGEMENT SYSTEM IMPLEMENTATION

The test building is modelled using EnergyPlus [2] and calibrated against real data from a full instrumented dwelling located in Ireland and described in [9]. This paper describes how to implement an EMS by using real-time smart meter data and home area network (HAN) as two-way communication. These two components together can provide electricity consumption data at 15 minutes time step resolution and they can also allow to control household appliances. In addition to improve positive economic results from the EMS, an energy price forecast, a load scheduler and an energy consumption monitor are used. The objective function of the energy manager system is to reduce occupants' cost using an electricity price scheme and set a comfort temperature band. Figure 2 provides an overview of the heating system controlled by the EMS. The two control data point used by the algorithm are the status (ON/OFF) of the circulation pump (HSE) and the heating storage set point temperature (HST). In Table 2 is reported the thermal comfort settings agreed with the building owner and used in the system as constraints. Such constraints are being satisfied for each simulation time step. Then in Table 1 is possible to find the electricity tariff used for the assessment of the system.

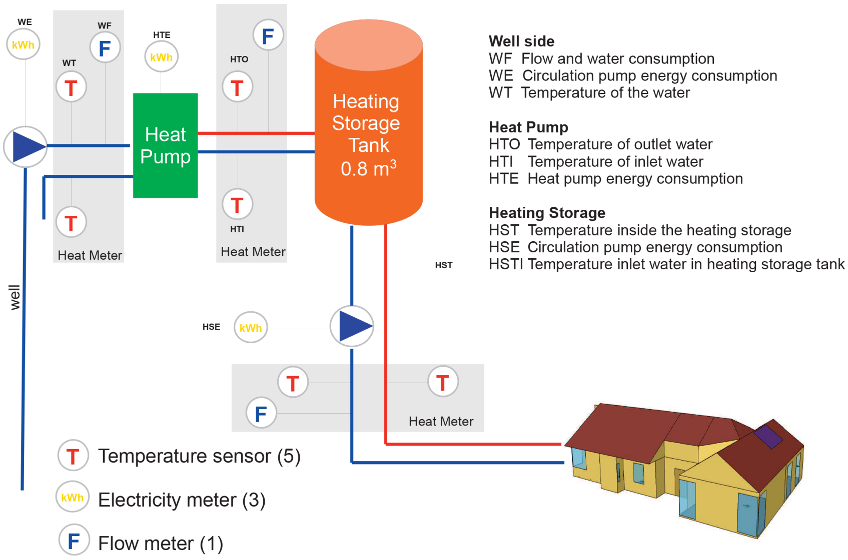

Figure 2: Heating system design

3.1 Approaches to develop an EMS

In general there are two main approaches for energy consumption management in buildings:

- Reducing consumption - can be achieved by raising awareness for more careful utilization patterns as well as building more energy-efficient buildings.
- Shifting consumption - can be done by shifting high-load household appliances to off hours in order to decrease the Peak to Average Ratio (PAR) [6] in load demand.

The former method is very common and easy to carry out, whereas, there are two major obstacles to implement the latter one. Firstly, there are not too many effective building automation systems [12]. And secondly it is not evident how the electricity price variations for end users can reduce residential electricity cost using automated controllers. Therefore, this and other studies has been ongoing to find effective way to shift loads. The EMS implementation developed is able to give an easy-to-access information on home electricity consumption in real time to control appliances and optimize the power usage at home [11]. The design of a proposed EMS will be described in the next section. In order to run the simulation, EMS has to change the status of the existing instance, which is regularly checked by the mediator component. Once mediator triggers that the instance status was updated, it will run the building simulation via BCVTB, which in turn runs the EnergyPlus. During the simulation all the obtained sensor results are sent and stored in the SimAPI database by BCVTB. Thus, EMS can get all these sensor results during the simulation and control the simulation by using API requests to SimAPI. So this process of EMS running and controlling the simulation is similar to other researches [5], where a scheduler component collects periodically reports from a monitor and a price predictor components, analyses all received data and decides an optimal choice for energy consumption scheduling, and controls all household appliances in form of on and off commands with specified power levels over either wired or wireless HAN.

3.1.1 SimApi Control Flow

Initially, the simulation start sending a begin request to SimAPI server. Once it is done successfully, the software program waits between fourteen an twenty seconds necessary to wait until Energy Plus simulation environment warms up. Since the simulation covers one month, 2881 timesteps will be generated during it. An interval between each timestep covers 15 minutes. The total simulation time is 43.200 minutes, which is equivalent to 720 hours or 30 days. At each time step the EMS retrieves the data from the sensor. If it gets new sensor information, then based on the control algorithm, the program either turns on or turns off the heating system by specifying the mentioned earlier min or max temperatures. To achieve it, the setting point of the thermal storage and the status of the circulation pump are sent to SimAPI server.

3.2 EMS - First Prototype

For the first version of EMS the logic in Figure 3 was used to control the heating system. The main goal of the system is to minimise energy consumption, maintaining the ambient comfort within the set points as Table 2 illustrates. The strategy to achieve this goal is to shifts the energy consumption of the heating system two hours before the peak time, charging the thermal storage. As shown in Figure 2 there are two control mechanism for the heating system: the Circulation Pump (HSE) and the set point temperature on the heating storage tank (HST). With regards to Circulation Pump when it is on or off, the heating system has the minimum temperature set at $40°C$, whereas the maximum set at $55°C$. The maximum temperature outlet of the heat pump is $60°C$ so taking into account the thermal losses from the storage tank, the effective maximum temperature set point

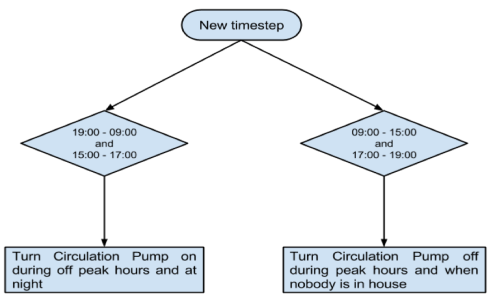

Figure 3: Control Flow First Prototype

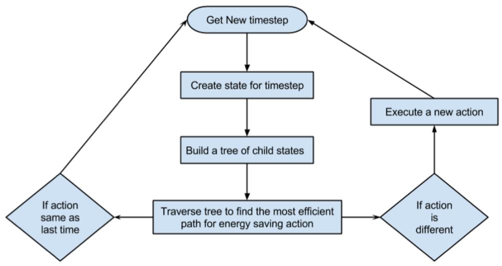

Figure 4: Control Flow Smart Algorithm

achievable is $55°C$. The minimum temperature set point is the result of a parametric analysis performed on the building model. This temperature allows the system to meet the energy demand also during the winter design days. These constant temperature values are stored in utilities class.

3.2.1 Algorithm description

The First Prototype creates and uses a single instance of an Hypertext Transfer Protocol (HTTP) client during its execution for all communication it needs. From Utility class, it requires most of the static variables such as temperature, server name and others. The program has two conditions based on the sensor reading at each timestep. The first condition checks whether the data point for the timestep was produced or not. The timestep counter is incremented only if a new data point is generated. Whereas the second condition checks if the program needs to turn off the heating system, and it is done by setting a status of circulation pump to OFF and the temperature at $40°C$. So the second condition is true only if time is 9 or 17, which represent time when there is no one at home and a start of peak hours respectively. The last condition is for checking whether the program needs to turn on the heating system, and it is done by setting the status of circulation pump to ON with the temperature at $55°C$. And it is true only if time is either at 15 or at 19, which represent two hours before peak price and off-peak hours respectively. To process results, all data points were retrieved from the SimAPI database during the simulation.

3.3 EMS - Smart Control

In this section is described a design and an implementation of a Smart Control Algorithm. The control algorithm depends on two elements. The first one is the data point retrieved from SimApi during the simulation and the second one is a data structure to calculate the optimal action for next timestep. The data structure is a binary tree and it has only two statuses to control the heating system: on or off. The system components are the following:

- Weather Forecaster. This module, given the current weather condition, forecasts the outside temperature and solar radiation for next timesteps.

- Predictor. It is a class that given the building status uses a data model to retrieve the optimal action for the next timestep to send to the simulation.

3.3.1 Weather Forecaster

This module forecasts the weather conditions for the next hours. The module could retrieve data from a weather API or provide an estimate temperature and solar radiation. The estimation is calculated retrieving the temperature of the previous hours and week and it will provide an average. The same calculation is used for the solar radiation.

3.3.2 Predictor

After an analysis of different machine learning techniques on the data set and evaluating them, M5P tree based model [10] was chosen due to its excellent error indicators. These models were produced in R language and were interfaced to the Java implementation through JRI [4]. The model input is outside temperature, inside temperature, solar radiation and time of the day. The output is the next 15 minutes temperature variation inside the building if the heating system is ON or OFF. Each created child state predicts delta temperature using M5P models built in R. In order to achieve that, R model class reads and uses two files that contain models for each status of the heating system when it is ON and OFF. Then the child state calls a method of the model class with all necessary parameters such as the parent state energy consumption of the heating pump, tank temperature, outside temperature, PV production and model type (either on or off). After that, this method is executed, and it returns the forecast of the ambient delta temperature for the child state, which can calculate using the tree the inside temperature and other necessary values [13].

3.3.3 Algorithm description

The controller, using the output of the predictor, checks whether a child state can keep the comfort temperature. If a generated state cannot guarantee the minimum comfort temperature, then it will be removed from the tree and will not be used during the tree traversal. Also, at the beginning of the tree construction, the system checks if the heating system OFF state child of the root violates the comfort temperature. In that case, a tree will not be build because the only valid child will be the ON state, so it means the heating system should be switched on. A tree traversal in this particular case will not be executed, because it is unnecessary to check only ON state child of the root. With regards to the tree traversal, it traverses until it reaches the end child states. For each node, the controller it sums up the energy consumption of each state, so that parent state will add the least energy consumed child state consumption to its own. And at the end it will return this tree of states with updated energy consumptions. Thus, root state can decide what to do next. The options are either to turn ON or turn OFF the heating system.

4. SYSTEM EVALUATION

The evaluation test of the smart controller and first prototype revealed that three seconds are necessary to send the next action to the system. So this interval gives a time gap for EMS to update settings or to do new actions. In the latter case, sensors take next measurements after few milliseconds a new action is added to the database. The controllers were tested on Windows 7 PC equipped with an Intel Core i3-2130 CPU at 3.40 GHz and 16 GB of RAM. From the computational resources point of view, First Prototype did not have any problem with physical memory and processing power, whereas the Smart Controller experienced a heap memory problem with R interface. Since JRI engine does not support multiple threads, at most one JRI instance can exist in a given JVM process. However, this feature causes two problems. Firstly, a deadlock can happen if more than one instance of JRI engine will be created. So the program or the thread calling the R engine object will be waiting forever to get the request back from it. Secondly, when a single thread of JRI is used, it will stay in Heap Space of the program till the end of the program. Consequently, the Java garbage collector cannot reclaim the space of objects associated with it. As a result, when the Smart Controller makes new predictions, the size of objects referenced by this thread will continuously grow every time and consume more memory. Over time, the leaked objects occupy all of the available Java heap space and cause Java heap space error. Thus objects inaccessible by the running system reside in main memory and cannot be collected in order to free the Heap space. Testing showed that the size of the data structure increases by 22 MB every ten calls to R models. Thus, this size without including a number of nodes becomes around 6 GB, whereas the server RAM (physical memory) was only 12 GB. Therefore, simulations with more than two nodes at each timestep will result in out of memory error.
With regards to nodes, using a simulation of one hour prediction, then it will perform at most 16 forecasts of delta temperature by using models built in R. Sixteen nodes is because, as explained in implementation details of the First Prototype, every 15 minutes a new timestep is generated, so 4 timesteps in a hour. Then at each timestep there are two possibilities such as turning on and turning off the heating system. Thus the total number of states that will be created at each timestep during the prediction is 2 to power of 4, which is 16. But Smart Controller will not create 16 nodes every time, since it prunes away nodes that have temperature outside the comfort level. Taking all these calculations and testing into account, for an evaluation part only a simulation with one hour prediction was performed. Scalability is a major issue for any system. The EMS was designed and tested to control an energy consumption of the single building model. In order to make this system more scalable, the application and testing of EMS for other different house models and an overall code review should be considered.

5. SIMULATION RESULTS

The First Prototype illustrated that by using a rule-based algorithm it is possible to reduce the energy consumption of the building by 25% and improve the efficiency of the energy demand. Also this rule-based EMS took into account the peak hours for energy price and the occupants' lifestyle, e.g. when they are at home or work. However simulation results in Figure 5 show that Smart Controller performed much better than both the Baseline Consumption and the First Prototype. Using advanced machine learning techniques and a forecaster, the developed Smart Controller reduces the energy consumption of the heating system by 50% and improve the efficiency of the energy demand, optimize the energy consumption.

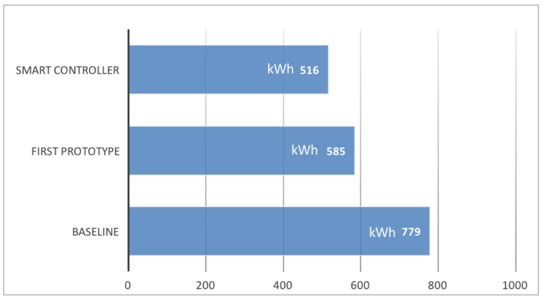

Figure 5: Electricity Consumption in three different cases for the simulation

Figure 6 shows the aggregated electricity consumption for the baseline and for the two controllers for the month of January 2015 using weather data from a nearby weather station. A 15 minute time resolution is used, where each data point is determined by summing each time associated electricity consumption value (kWh) for that data point (30 instances) for the entire month. The PV contribution, is determined in a similar way, but is subtracted from the consumed electricity to allow a net value to be determined.
Only during the day, the First Prototype consumes less energy. During the evening, it increases the consumption to restore the storage tank temperature. From Figure 6 is clear that the baseline consumption differs from simulation results constantly during the day. Especially from 10:15 am till 17:00, the system control flow logic turn off the heating system when nobody is at home, reducing the consumption. On the other end, the smart controller optimizes the energy consumption during through the day reducing the peaks. This strategy leads to an overall reduction also compared to the First Prototype.

6. CONCLUSIONS

The present paper reveals that the performance of an all-electric dwelling subject to heuristic optimisation algorithms, can improve overall system performance and contribute to a significant electricity consumption reduction. The overall building, with its control systems, display a 25% energy reduction than the baseline case using a rule based algorithm. Furthermore, the paper highlights an increase energy reductions up to 50% obtained with a smart control algorithm that uses a statistical model to forecast the temperature variation inside the building. It will be the subject of further research in the area to design an algorithm that can progressively learn from the user habits and occupancy profiles of each room using more advanced data structures and strategies. Moreover, this work considered only an energy consumption of a building heating system. However, for fu-

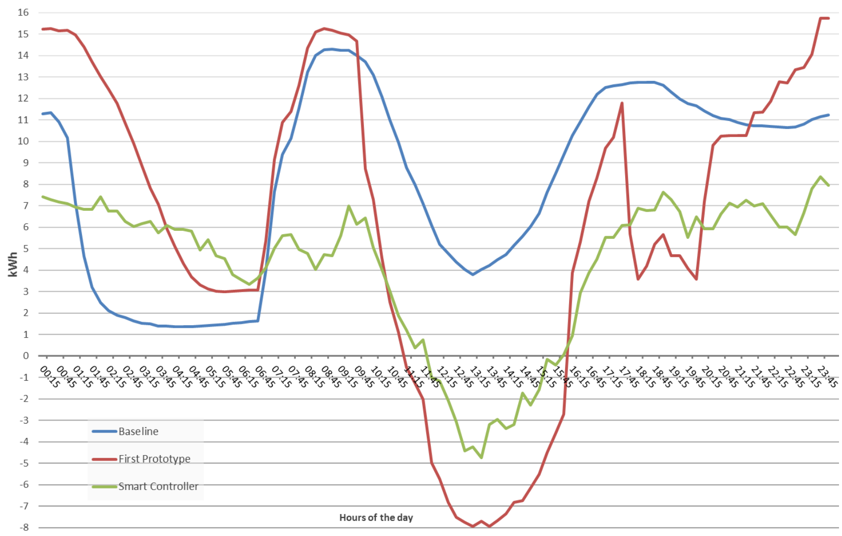

Figure 6: Cumulative Energy Consumption profile in the three cases

ture work, learning and analysing energy usage of other main household appliances, e.g. air conditioner and refrigerator, can be done.

Acknowledgments

This work was conducted in the Electricity Research Centre, University College Dublin, Ireland, which is supported by the Commission for Energy Regulation, Bord Gais Energy, Bord na Mona Energy, Cylon Controls, EirGrid, Electric Ireland, Energia, EPRI, ESB International, ESB Networks, Gaelectric, Intel, SSE Renewables, and UTRC. This report come from research conducted with the financial support of Research in Third-Level Institutions (PRTLI) under Grant Number [R12681]. Special thanks to Mr. Brendan Robinson, whose dwelling was use as experimental test bed for this research.

7. REFERENCES

[1] Z. Conka, P. Hocko, M. Novak, M. Kolcun, and M. Gyorgy. Impact of renewable energy sources on stability of ewis transmision system. In *Environment and Electrical Engineering (EEEIC), 2014 14th International Conference on*, pages 75–79, May 2014.

[2] P. G. Ellis, P. A. Torcellini, and D. B. Crawley. *Simulation of energy management systems in EnergyPlus*. National Renewable Energy Laboratory, 2008.

[3] P. Eriksen, T. Ackermann, H. Abildgaard, P. Smith, W. Winter, and J. Rodriguez Garcia. System operation with high wind penetration. *Power and Energy Magazine, IEEE*, 3(6):65–74, Nov 2005.

[4] E. J. Harner, D. Luo, and J. Tan. Javastat: A java/r-based statistical computing environment. *Computational Statistics*, 24(2):295–302, May 2009.

[5] A.-H. Mohsenian-Rad, V. Wong, J. Jatskevich, and R. Schober. Optimal and autonomous incentive-based energy consumption scheduling algorithm for smart grid. In *Innovative Smart Grid Technologies (ISGT), 2010*, pages 1–6, Jan 2010.

[6] A.-H. Mohsenian-Rad, V. W. Wong, J. Jatskevich, and R. Schober. Optimal and autonomous incentive-based energy consumption scheduling algorithm for smart grid. In *Innovative Smart Grid Technologies (ISGT), 2010*, pages 1–6. IEEE, 2010.

[7] G. M. Nijssen and T. A. Halpin. *Conceptual Schema and Relational Database Design: a fact oriented approach*. Prentice-Hall, Inc., 1989.

[8] F. Pallonetto, E. Mangina, D. Finn, F. Wang, and A. Wang. A restful api to control a energy plus smart grid-ready residential building: demo abstract. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, pages 180–181. ACM, 2014.

[9] F. Pallonetto, S. Oxizidis, and D. Finn. Exploring the demand response potential of a smart-grid ready house using building simulation software.

[10] J. R. Quinlan et al. Learning with continuous classes. In *5th Australian joint conference on artificial intelligence*, volume 92, pages 343–348. Singapore, 1992.

[11] D. Ren, H. Li, and Y. Ji. Home energy management system for the residential load control based on the price prediction. In *Online Conference on Green Communications (GreenCom), 2011 IEEE*, pages 1–6, Sept 2011.

[12] A. Schumann, J. Ploennigs, and B. Gorman. Towards automating the deployment of energy saving approaches in buildings. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, BuildSys '14, pages 164–167, New York, NY, USA, 2014. ACM.

[13] Y. Wang and I. H. Witten. Induction of model trees for predicting continuous classes. 1996.

[14] M. Wetter. Co-simulation of building energy and control systems with the building controls virtual test bed. *Journal of Building Performance Simulation*, 4(3):185–203, 2011.