On Sensor Data Simulation

Alexandru-Ciprian Zăvoianu, Ciprian-Dorin Crăciun

Research Institute e-Austria,
blvd. C.Coposu, 4, room 045B,
300223, Timişoara, Romania
Tel/Fax 00 40 256 244834
e-mail: {zavoianu.ciprian, ccraciun}@info.uvt.ro

***Abstract*—In the context of the *DEHEMS* project, aimed at analyzing real-time sensor readings related with the energy performance of individual households, the research and development stages of various modules have underlined the need for a fast and scalable means of generating large amounts of high quality sensor data. This paper introduces a multi-purpose sensor simulation system (*DS3*) that successfully meets all the data generation requirements imposed by *DEHEMS*. Primarily designed as an internal tool for testing and tuning the storage and data analysis engines, *DS3* will act also as a support platform for a number of end-user orientated applications.**

***Index Terms*—sensor simulation software, quality simulation data, spot energy consumption profiles.**

I. Introduction

The *Digital Environmental Home Energy Management System* (*DEHEMS*) [5] is an European Union funded project aimed at acquiring, storing and analyzing real-time information regarding the overall energy performance (electric, thermic, gas) of individual households and services. The purpose of the project is to improve current energy consumption monitoring approaches with an overall goal of helping to reduce CO_2 emissions across Europe.

The philosophy behind *DEHEMS* is to move beyond classical energy input models, that only monitor the levels of energy being used, to energy performance models which also take into consideration the way energy is being used. In order to achieve this, the system will require access to real-time sensor data in areas such as household heat loss and individual appliance performance as well as general energy usage monitoring information. By providing individual feedback and energy efficiency recommendations for each household, *DEHEMS* will have a major social impact by helping to personalize action on climate change. Several studies regarding the effect of real-time feedback on home energy use have been carried out and they have shown that energy consumption feedback can lead to savings between 5 and 20% [4]. These studies also pointed out that highly personalized and detailed feedback has the strongest impact on household energy savings.

Google PowerMeter [8] is a solution prototyped by Google Inc. aimed at helping users monitor and reduce their household electric energy consumption. The main objective of this project is to provide individual users with access to detailed real time data regarding personal electric energy use. One major difference between *Google PowerMeter* and *DEHEMS* is the fact that whilst the focus of the latter is to analyze electric energy consumption within the broader context of overall individual household energy performance models, the former is designed as a platform that will support several future online electric energy information services.

The current development stage of the storage engine employed by the *DEHEMS* project requires a means of rapidly generating and flooding the storage engine with large amounts of sensor data in order to simulate the data storage requirements that would be imposed by the real-life hardware energy monitoring system. The research stage of the data-analysis engine has emphasized the need for a system that could generate very accurate sensor data, preferably based on previously defined energy consumption patterns. Furthermore, in order to test and help adapt *DEHEMS* to the various types of hardware power meters available, the sensor simulation application had to be able to export the simulated data in various formats like *XML*, *JSON* and other custom formats, that were generically named aggregated data packages (ADPs).

Additionally, hardware constraints were also imposed on the *DEHEMS Sensor Simulation System (DS3)*. The system had to be scalable in order to take maximum advantage of the underlying hardware architecture and was required to perform well on standard PCs. The goal was to achieve an individual PC insertion rate of over 4.200 ADPs / second. Taking into account the fact that the individual household hardware monitoring system provides updates every 6 seconds, this means that a single PC should simulate the real-time output of $\simeq 25.000$ households.

Our first intention was to adapt an existing sensor simulation system such as to satisfy the requirements imposed by the *DEHEMS* project. We looked at systems capable of simulating and operating with large amounts of sensor data. Such types of simulators abound in *Wireless Sensor Network* (*WSN*) literature. Some of these *WSN* simulators, like *ns-2*, *SensorSim*, *SENS* [10], and *EmStar [7]* are in fact simulation frameworks that allow users to build complex custom simulation applications on top of them. They exhibit a lot of useful features for *WSNs* simulation applications like a high level of abstraction, modular architecture, customizable communication protocols and feedback modules. After a careful analysis of these frameworks, we have decided that adapting/using such a framework for the *DEHEMS* project is not a viable solution. This is due to the fact that these *WSNs* simulation frameworks are primarily orientated on detailed modeling of the real-life *WSN* communication system behavior (distances, obstacles, signal strengths, routing protocols, transmission errors, and others) rather then on generating and aggregating specific high-quality sensor data. Preliminary tests have shown that the inherent overhead of using such a complex framework in order to support our sensor simulation application would have imposed drastic speed penalties.

The lack of a flexible sensor simulation platform that could be adapted in order to satisfy all the data quality, quantity and format constraints imposed by the *DEHEMS* project motivated the need to develop a new sensor simulation system that could exploit project specific details in order to achieve the imposed performance levels.

The remainder of this paper is organized as follows. *Section II* describes the data simulation techniques. *Section III* covers the architecture of *DS3*. *Section IV* presents some of the results obtained using the simulation system. Finally, *Section V* concludes and offers an overview of future work regarding *DS3*.

II. Data simulation techniques

The *DEHEMS* system will collect large amounts of real-time sensor data from individual households. For now, the system receives data from at most 12 sensors per household, each of these sensors having a sample rate of 10 readings per minute. These 12 household sensors are divided as follows:

- *one general household electric meter* which provides information regarding the spot general electric power consumption of the entire household;
- *one temperature sensor* which reports the household temperature (better quality data can be provided by having more sensors around the house and locally computing an average of their reported readings);
- *one gas meter* which reports the spot gas consumption of the household;
- *up to nine individual electric sensors* which report the spot power consumption of various individual appliances in the household;

The combined readings of the 12 sensors mentioned above make up the spot energy consumption profile of the household.

In the current stage of development, the *DEHEMS Sensor Simulation System* contains two distinct stages of generating spot energy consumption profiles: *gross direct simulation* and *detailed pattern based simulation*.

A. Gross direct simulation

In this first stage, basic sensor readings are simulated by estimating values based either on real-life measurements or data from studies in the filed of household energy consumption [6], [9], [1].

For instance, the spot power consumption reported by various individual appliances is estimated by taking into consideration the active power consumption (i.e. the average power consumption of the appliance) and the measured standby power consumption (if the appliance has a default standby mode). In ***Table I*** we present the active and standby power consumption levels of some of the appliances featured in *DS3*.

B. Detailed pattern based simulation

In order to accurately simulate complex readings like the one produced by the general household electric meter, we have identified the need to use energy consumption patterns.

In their simplest form, these patterns show the consumption state (active, standby, or offline) of every appliance in the house for a period of 24 hours. By combining these data with information regarding the power consumption of individual appliances we can obtain, through aggregation, an accurate picture of the general household spot electric power consumption at a given time. The electric aggregation operation consists

Table I
POWER CONSUMPTION DATA FOR 20 HOUSEHOLD APPLIANCES

Household appliance	Active power (W)	Standby power (W)
Air conditioner	3250	15
Blender	300	0
Color TV	90	20
Computer (laptop)	30	5
Computer (PC)	90	10
Computer speakers	80	10
Economical light bulb	30	0
Electric blanket	300	0
Electric mower	1500	0
Electric fan	200	0
Fax machine	8	8
Iron	1000	30
Laser printer	45	10
Microwave	1000	10
Refrigerator	800	0
Standard light bulb	100	0
Stereo CD player	20	5
Toaster	850	0
Vacuum cleaner	730	0
Washing machine	500	0

of simply adding up all the spot power consumption readings from the individual device sensors.

More complex energy consumption patterns can be constructed using additional statistical data regarding the average number of occupants / house, the household occupancy pattern and the ownership level of each appliance, as shown in [11].

In ***Figure 1*** we show how the direct sensor readings are used within *DS3* to construct an spot household energy consumption profile: $Ia_1, Ia_2...Ia_n$are the individual appliance sensor readings, $T_1, T_2...T_m$ are the temperature sensor readings, E is the reading from the general household electric meter, T is the average household temperature reading, G is the gas meter reading and P is the obtained spot energy consumption profile.

While all the individual appliance sensors aid in simulating the reading produced by the general electric meter, only nine of them (selected using predefined filters) contribute individually to the consumption profile. The appliance selection filters take into consideration aspects like:

- *appliance power consumption* - select the appliances with the highest power requirements;
- *appliance usage* - select the appliances that are most frequently used;
- *personal preferences* - select appliances based on *DEHEMS* user preferences;

The temperature aggregation operation is based on a weighted function as one may want to give more relevance to temperature readings in some parts of the house – like bedroom, living-room – over readings from other parts – such as garage, attic.

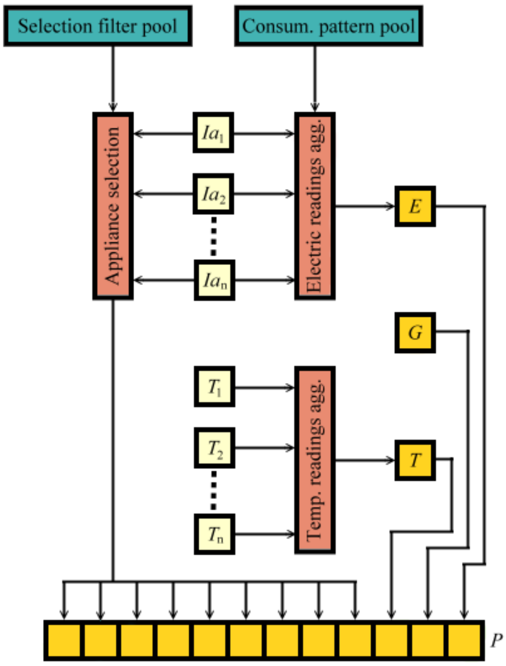

Figure 1. Framework for generating the household energy consumption profile in *DS3*

III. THE ARCHITECTURE OF THE SYSTEM

DS3 is constructed around a very simple *Event-Driven Architecture* (*EDA*). The reason we have chosen this model is the fact that we wanted our system to mimic, as close as possible, the structure of the real-life hardware energy monitoring system which is primarily composed of loosely coupled components (meters, sensors, gateways). While the meters, sensors and gateways in a given household constantly need to exchange messages, there is no need for communication between sets of *DEHEMS* energy monitoring components situated in different households, as each set sends its ADPs directly to the *DEHEMS* storage engine.

Structural similarity to the real-life monitoring system is required in order to allow *DS3* to simulate communication and data acquisition errors, hardware failures and dynamic appliance selection filters.

Moreover, the flexibility provided by an event-driven architecture enables the rapid transformation of *DS3* into a hybrid simulation platform in which part of the sensor readings come from real-life sensors which smoothly interact with the rest of the system.

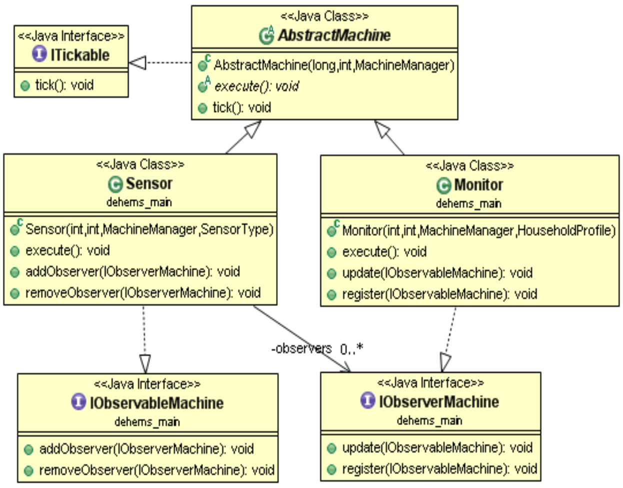

Figure 2. DS3 core structure

The *Java platform* was chosen for our first implementation of *DS3*, as we wanted to develop a cross-platform application and future development plans are related with creating user oriented web applications based on simulation applets.

A. Core structure

The core of *DS3* is made up of a set of Java interfaces and classes acting accordingly with the architecture presented in ***Figure 2***:

- The central *ITickable* interface helps to implement the *EDA* pattern by providing a single *tick* event , and each implementer (sensor, monitor, and exporter) executes its main task only inside this call;
- Several sensor classes provide the logic for the *gross direct simulation* stage and (with a frequency of 20 times per minute) output the values that would be produced by individual appliance sensors or by basic temperature or gas sensors;
- Additional monitor classes provide the logic for the *detailed pattern based simulation* stage, where their role is largely similar to that of modern real-life household energy meters like the *Current Cost 128* [3]; they collect data from individual appliance sensors and (with a frequency of 10 times per minute) use the data to construct the spot household energy consumption profile; the only difference is that, unlike real meters, they do not have access to a general household electric consumption sensor (the *Current Cost* clamp) and as such, they have to accurately simulate these sensor readings using predefined consumption patterns;

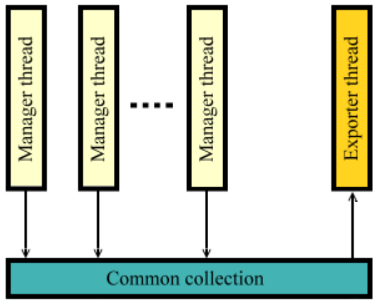

Figure 3. *DS3* multithreading schema

- Also, there are interfaces that help construct the communication channel (a classical *observer pattern*) between the sensors and the monitors; we have opted for a bottom-up communication model in which the sensors push the data onward to the monitors which observe them;
- Finally, there is a need for some classes that export the aggregated (sensor reading) packages in specific formats needed by various external storage and analysis engines.

The core structure described above encapsulates the entire logic needed to generate spot energy consumption profiles.

B. Multithreading mechanism

In order to achieve the design imperative of taking maximum advantage of the underlying hardware architecture, *DS3* employs a multithreaded consumer-producer design pattern (***Figure* 3**). Two types of threads exist in *DS3*:

- multiple *manager threads* which have the role of producing as many ADPs as possible (producers);
- a single *exporter thread* which has the role of forwarding the ADPs via network to the *DEHEMS* storage engine (consumer);

The interaction between threads is kept to a minimum in order to increase efficiency. Each manager thread holds individual pools of sensor and monitor objects which only need to interact with objects hosted in the same thread in order to produce ADPs. When in an active state, the only action required from a manager thread is to continually trigger the *tick()* event on all of its hosted sensor and monitor objects.

Data sharing between the producer threads and the consumer thread is realized via a common queue fitted with a bounding mechanism.

Scalability is very easy to achieve using this multithreaded architecture with the sole requirement of choosing the number of manager threads according to the physical architecture of the computer. The best test results have been obtained when the number of manager threads exceeded by one the number of processing cores of the computer.

Need be the case, *DS3* can easily be adapted in order to accommodate multiple exporter threads, but for systems with up to 8 cores, one exporter thread has proven to be sufficient.

C. DS3 I/O

The input for *DS3* consists of individual appliance sensor simulation data and sets of energy consumption patterns. In the current implementation both are stored as Java properties files.

This is an example for a properties file that defines an individual sensor monitoring a TV set:

```
ID = 2
description = Brand name#Model number
unit = watt
activeValue = 90
standbyValue = 20
```

Here are 4 entries from a set of energy consumption patterns that shows when individual devices in the house are active (similar sets exist for showing when devices are in standby mode). The key is the individual device ID and the value is an array of time intervals in which the device is active.

```
1 = 18:20:00#19:50:00
4 = 00:00:00#23:59:59
2 = 07:50:00#08:10:00#14:00:00#14:30:00
8 = 00:00:00#23:59:59
```

Each household energy consumption profile the system generates is wrapped in a *XML*, *JSON* or custom format (ADPs) depending on the requirements of the simulation scenario. Here is an example of an *XML* wrapping used to stress test the storage engine:

```
<msg>
  <date>
    <dsb>00030</dsb>
    <h>00</h><m>20</m><s>11</s>
  </date>
  <src>
    <id>00077</id>
  </src>
  <gene>
    <watts>168</watts>
  </gene>
  <tmpr>
    <celsius>24.8</celsius>
  </tmpr>
  <gas>
    <m3>0.008<m3>
  </gas>
  <ia1>
    <watts>46</watts>
  </ia1>
  ...
  <ia9>
    <watts>100</watts>
  </ia9>
</msg>
```

IV. System performance

Our intent is to install *DS3* on several desktop computers in our laboratories in order to stress test the *DEHEMS* storage engine. The quantity of simulated real-time output data should match the one that would normally be forwarded by a fully working real-life hardware energy monitoring system covering over 1.000.000 households.

In order to accomplish this goal using 40 PCs, *DS3* has to achieve a data generation and insertion rate of approximately 4.200 aggregated data packages / second on each individual computer. A series of tests were conducted on a personal computer with a dual core AMD Athlon64-LE-1640 CPU clocked at 2600 Mhz and 2Gb of RAM with a speed of 800 Mhz.

The simulations used input data from a pool of 50 individual appliance sensors and a pool of 10 sets of energy consumption patterns each containing at least 15 entries. The application was set up to use three manager threads.

We ran tests for five different time intervals: 1 minute, 5 minutes, 10 minutes, 20 minutes and one hour. We allowed for a 5 second setup time for each test and we repeated the test 10 times for each time interval.

The results we have obtained are summarized in ***Table II*** and show that the current implementation is more than capable of delivering the amount of data required by our storage engine tests. Even more, the small values of the coefficient of variation (c_v) and the near constant insertion speed show the fact that the data simulation technique is sound and that the implementation exhibits a good level of stability.

In order to compare the quality of the simulated data we have charted, as shown in ***Figure 4***, the general power consumption reported by the current *Current Cost* [3] meter in a 30 minutes test interval in which we precisely

Table II
DS3 DATA INSERTION TESTS SUMMARY

Time intervals (seconds)	65	305	605	1205	3605
Average inserts	332,094.40	1,678,241.70	3,409,745.90	6,751,796.10	20,307,480.30
Average inserts per second	5,109.14	5,505.71	5,635.94	5,603.15	5,633.14
σ (for inserts per second)	147.34	168.20	160.33	193.11	172.54
c_v (for inserts per second)	0.0288	0.0305	0.0284	0.0344	0.0306

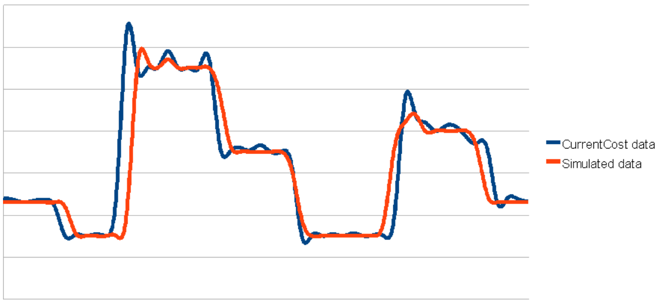

Figure 4. Simulated and real-life data comparison

monitored household appliance usage. We then used our observations to construct a simulation scenario that could mimic the real-life experiment. We have also charted our results for the spot household consumption profile.

The results show that, if provided with quality input data, our two stage simulation technique can generate very accurate consumption readings that closely resemble the one produced by real-life hardware monitoring system.

V. Conclusions and future work

The sensor simulation system we have described in this paper has successfully achieved its primary design goals by proving that it is able to generate very large amounts of quality sensor data that is used to develop the storage engine and the data analysis engine of *DEHEMS*.

Future work regarding *DS3* aims to replace the gross sensor data (active consumption and standby consumption) that characterize individual appliance sensors with more realistic individual consumption profiles that exactly map the consumption behavior of these appliances (spikes, low consumption periods, gradual shutdown) during their active and standby periods (see [2]).This will enable the creation of even more realistic spot household consumption profiles.

Another future development direction consists of developing a means of automatically deploying *DS3* in a distributed computing environment (node communication and basic synchronization). This would enable the user of the application to easily create complex simulation scenarios that involve several hundreds of thousands of households.

In order to be able to accommodate very large simulation scenarios that also require a high level of data quality, *DS3* will have to be fitted with a high number of accurate individual consumption patterns. As even the process of creating one such accurate consumption pattern is a tedious task for a human, the development of an automatic means of generating such patterns is of great priority. Population based techniques are a very natural approach to this problem. For example, the pattern generation method could be based on a genetic algorithm which would require some basic input like: the average number of occupants / household, the household occupancy pattern, the ownership level of each appliance and a large pool of distinct individual appliance models. The fitness function should check the degree in which an individual (i.e. a candidate pattern) can output a household energy consumption profile that is *similar* to a measured, real-life, consumption profile stored in a benchmarking test set. The main advantage of this method is the fact that the obtained set of patterns (and implicitly, profiles) will be quite diverse and realistic. The downturn consists in the relative high complexity and slow speed of the method.

Future work regarding *DS3* is also related to exposing some of the capabilities of the system as web services (SOAP based services). This will help transform *DS3* into a simulation framework that could easily provide support for 3rd party applications like household simulation applets.

Acknowledgments

This work is supported by the *EU* through the *STREP* project *DEHEMS* (*FP7* reference number 224609) and by the Romanian *DEHEMS-RO* project *RO_PNII_20 EU/11.06.09*.

REFERENCES

[1] Center for Climate Change and Sustainable Energy Policy of the Central European University, “Appliance Standby Power Consumption Store Survey 2008”, Czech Republic, 2008.

[2] D. Cogan, M. Camilleri, N. Isaacs, L. French, “National Database of Household Appliances – Understanding Baseload and Standby Power Use”, Proceedings of the 4th International Conference on Energy Efficiency in Domestic Appliances and Lighting (EEDAL’06) 21-23 June 2006. London, UK.

[3] Current Cost 128 specifications, http://www.currentcost.com/product-cc128-specifications.html

[4] S. Darby, “The effectiveness of feedback on energy consumption”, 2006.

[5] DEHEMS project homepage, http://www.dehems.eu/

[6] A. S. Fung, A. Aulenback, A. Ferguson, V. I. Ugursal, “Standby power requirements of household appliances in Canada”, Energy and Buildings 35 (2003) 217–228.

[7] L. Girod, J. Elson, A Cerpa, T. Stathopoulos, N. Ramanathan, D. Estrin, “EmStar: a Software Environment for Developing and Deploying Wireless Sensor Networks”, Proceedings of the USENIX Technical Conference, Boston, 2004.

[8] Google PowerMeter, http://www.google.org/powermeter/

[9] J.P. Ross, A. Meier, “Whole-House Measurements of Standby Power Consumption”, Proceedings of The Second International Conference on Energy Efficiency in Household Appliances, Italy, 2000.

[10] S. Sundresh, W. Kim, G. Agha, “SENS: A Sensor, Environment and Network Simulator”, Proceedings of 37th Annual Simulation Symposium, 2004, pp. 221–230.

[11] R. Yao, K. Steemers, “A method of formulating energy load profile for domestic buildings in the UK”, Energy and Buildings 37 (2005) 663–671.