

Energy flexometer

Citation for published version (APA):

Babar, M., Grela, J., Zadowicz, A. O., Nguyen, P. H., Hanzelka, Z., & Kamphuis, I. G. (2018). Energy flexometer: transactive energy-based internet of things technology. *Energies*, 11(3), Article 568.
<https://doi.org/10.3390/en11030568>

DOI:

[10.3390/en11030568](https://doi.org/10.3390/en11030568)

Document status and date:

Published: 25/02/2018

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy



If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Article

Energy Flexometer: Transactive Energy-Based Internet of Things Technology

Muhammad Babar ^{1,2,*} , Jakub Grela ^{1,*}, Andrzej Ożadowicz ^{1,*} , Phuong H. Nguyen ², Zbigniew Hanzelka ¹ and I. G. Kamphuis ²

¹ Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering, AGH University of Science and Technology, Krakow 30-059, Poland; hanzel@agh.edu.pl

² Electrical Energy Systems Group, Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven 5612AJ, The Netherlands; p.nguyen.hong@tue.nl (P.H.N.); i.g.kamphuis@tue.nl (I.G.K.)

* Correspondence: M.Babar@tue.nl (M.B.); jgrela@agh.edu.pl (J.G.); ozadow@agh.edu.pl (A.O.)

Received: 9 February 2018; Accepted: 3 March 2018; Published: 6 March 2018

Abstract: Effective Energy Management with an active Demand Response (DR) is crucial for future smart energy system. Increasing number of Distributed Energy Resources (DER), local microgrids and prosumers have an essential and real influence on present power distribution system and generate new challenges in power, energy and demand management. A relatively new paradigm in this field is transactive energy (TE), with its value and market-based economic and technical mechanisms to control energy flows. Due to a distributed structure of present and future power system, the Internet of Things (IoT) environment is needed to fully explore flexibility potential from the end-users and prosumers, to offer a bid to involved actors of the smart energy system. In this paper, new approach to connect the market-driven (bottom-up) DR program with current demand-driven (top-down) energy management system (EMS) is presented. Authors consider multi-agent system (MAS) to realize the approach and introduce a concept and standardize the design of new Energy Flexometer. It is proposed as a fundamental agent in the method. Three different functional blocks have been designed and presented as an IoT platform logical interface according to the LonWorks technology. An evaluation study has been performed as well. Results presented in the paper prove the proposed concept and design.

Keywords: transactive energy; demand response; energy management system; Internet of Things; smart metering

1. Introduction

With the growing implementation and use of distributed energy resources as well as renewable energy sources (RES) in power systems [1,2], the importance of effective energy management systems (EMS) with active control and monitoring functions has never been so high. The modern EMS organized with distributed control systems as well as building automation and control systems (BACS) provide tools for easy implementation of demand response (DR) and active demand side management (DSM) systems [3,4], the key mechanisms considered in effective energy and power management within the smart grid (SG) [5,6]. The SG concept has being proposed along last few years to modernize and facilitate the operation of power systems in the presence of the DER and RES, electric energy storages and local microgrids including prosumers. Relatively new concept in effective management of energy sources and loads connected to the SG is a transactive energy. Transactive energy (TE) has been introduced by the GridWise Architecture Council and refers to use of a combination of economic and control techniques to improve SG reliability and efficiency, using value as a key operational parameter [7,8]. Since control techniques are elements of the TE

definition, this fact determines their use in new fields as a part of the SG management strategy and takes all control, monitoring and energy management systems implemented in various applications (like homes, buildings, microgrids, substations etc.) to a new level. Despite novelty of the TE concept, it is becoming more and more popular and has being adapted in different control and monitoring applications [8,9]. Several papers discuss various approaches, technologies, tools as well as information and communication technologies could be used and in the TE [10,11]. According to [8,12], the TE concept is perceived as natural continuation and development of the DSM and EMS, especially in the context of providing a balance between electrical power supply and demand at the consumers and prosumers level. As it has been mentioned before, besides control techniques the value is a key operational parameter in the TE. In [13] Babar et al. propose an electrical energy price elasticity of demand, that can be used as a value for bids in electrical energy management. Moreover, in [14,15] the electrical energy price is applied as a control signal for an event-driven energy management concept and this way the price-based control mechanism can be considered as a part of the TE concept.

In the SG various sensors, actuators, meters and controllers are integrated in the power system equipment (distributed in power stations, distribution and transforming stations etc.) [16,17] and connected to the communication networks, allowing them to exchange data in real-time to control power generation and demand, forecast power consumption as well as update operational set-points, diagnose problems [18,19]. Different communication technologies and standards are used in the SG applications but taking into account new challenges related to the DR and DSM mechanisms implementation within the EMS, a new data communication platform should be unified and standardized for these applications. A relatively new paradigm in this field is an Internet of Things (IoT) based on the most popular communication protocol - Internet Protocol (IP). The IoT with its new IPv6 version is a network of various physical objects (nodes) connected to the Internet. These nodes contain their own embedded technology to interact with their external environment and internal states [20]. In this sense, the SG infrastructure can be classified as an Internet of Thing (IoT) application, since all these sensors, computers, actuators, meters and software agents mentioned earlier should working together as IoT nodes, enabling data exchange between themselves and providing tools to analyze, decide, and control devices individually.

Bearing in mind all the mentioned concepts, tools and technologies, it is possible to propose and implement market-based control mechanism (MCM) to dynamic manage energy supply and demand. It requires integration of distributed SG and BACS components with advanced demand and energy management tools. A Multi-agent System (MAS) paradigm is advocated as a useful and promising tool for advanced control applications and decentralized management [21]. Generally, the MAS is a network of two or more intelligent control and/or monitoring units (i.e., referred as agents) [22]. Agents are capable of interacting with each other in the control network, as well as they can be organized in multiple ways such that a global objective of the system should be distributed among all agents into a set of smaller tasks [4,23]. In context of EMS, every agent has the ability to monitor energy usage, primary process parameters (e.g., occupancy, comfort level) and control signals during an operation [14,15]. Agent is also equipped with communication module for sharing of data and information with other agents within the network [24,25].

During the last decade, the MAS has been widely considered in various power system applications and projects. For example, the PowerMatcher developed by the Netherlands Organization for Applied Scientific Research is a DR that balances demand and supply over local basis. A bidding mechanism is applied to manage the loads with more precision and efficiency. Devices inside the PowerMatcher are represented by agents, where they are organized in co-tree fashion. Each agent talks to upper-stream agent and expresses its willingness to consume or produce energy in the form of a simple bid (a demand or supply relationship). Based on the bids, the upstream agent (namely concentrator) decides what any device should produce or consume in order to keep the system balance [22,26]. A similar product based on the MAS methodology and referred as Intelligator has been developed by the energy department of Flemish Institute for Technological Research (VITO). Then, the VITO focused on the

development of a software library contains advanced algorithms, which enhance the intelligence of agents. In particular, the algorithms consider local electronic auction to regulate the system where participants of auction send their requested power in form of bid to an upstream-agent (namely auctioneer). The auctioneer finds a balance between production and consumption by adding all bids from agents. Then it responds to the participants with a demand schedule over day-ahead basis [27]. There are also some technological applications like PowerRouter by Nedap or Intelliweb by Mastervolt. These solutions perform an intelligent control of solar energy at home in order to increase power injection to the grid. They provide access to data via their data server to acquire and control in real-time [28,29]. In addition to these works and applications, there are other projects that use MAS based EMS with DSM to improve quality and control of power system, like ForskEL in Denmark [30]. Some of them are focused on local microgrids with prosumers and their collaboration with the SG [31,32]. Others propose and analyze possibilities of use of the BACS, Building energy management system and Home Automation Systems to organize distributed and integrated network platforms for effective implementation of the MAS for EMS and DSM applications [33,34].

Moreover, an ongoing the European Union project titled “Multi-agent systems and secured coupling of Telecom and EnErgy gRIDs for Next Generation smart grid services” is aimed at developing an IoT platform as a tool for low-voltage power grids management, control and monitoring. The project proposes technical solutions both for increasing the security of bi-directional communications as well as integration of last mile connectivity with distributed optimization technologies [35,36].

Despite the fact that many studies and research are carried out in the field of demand management performance in power system, current evidence is insufficient to provide and generalize principles. Moreover, different research communities tried to tackle the issue of DR integration in EMS within their own expertise, at the cost of precision in respective domain. Hence the studies are not representative, which is taken into consideration in this paper. The paper proposes “Energy Flexometer” as a monitoring, controlling and bi-directional communicating node or agent that can be integrated easily in a MAS-based EMS. Authors describe a concept of their solution and propose application interface for the universal IoT platform. Moreover, the Energy Flexometer as agent node has been implemented in a small proof-of-concept application to test and verify the concept and proposed communication BACS and IoT technologies. Scalability of the proposed solution is not considered in this paper. It will be a subject of future works.

The rest of this paper is organized as follows. The Section 2 provides details about design and concept of the proposed Energy Flexometer with its standardized logical interface. The Section 3 presents the demand elasticity estimation approach and algorithms. In Section 4 physical implementation of the proposed Energy Flexometer concept and technical solution is discussed. Details of a small test installation and the evaluation of results are discussed in Section 5. Finally, Section 6 gives the conclusions and future works.

2. Design and Concept of Energy Flexometer

2.1. Concept

The concept of energy flexometer lies in the implementation of TE at the low-voltage level, as shown in Figure 1. As in Figure 1, it can be observed that in TE based EMS or BACS, the aggregator and domotics are the smart nodes that distribute decision-making task among each other. In MAS based TE, as discussed in [22], the aggregator is the uppermost agent that has objectives to either mitigate network issues or solve local imbalance or both. On the other hand, domotic agents are coordinators transmits the aggregated value-proposition (i.e., bid) and receives control information from the aggregator in a real-time environment. Moreover, domotic agent standardizes the value-proposition as a key element of FlexiblePower Application Infrastructure (FPAI) platform—proposed and developed by Flexible Power Alliance Network. FPAI has introduced a set of rules and protocols to create interoperability between the aggregator and the domotic agents. However, there was not any

suitable effort to improve the interoperability between the domotic agents and the real physical loads/generators. Therefore, the concept of energy flexometer has been introduced in this study to fill in the research gap. It provides a standardized design of lower agents to make the architecture more interoperable. The concept of flexometer will help the entire architecture by three means. Firstly, the standardized embedded system of the flexometer can turn the dumb loads into the smart appliances, hence increasing more flexible demand in an aggregator portfolio. Secondly, the standard design will allow the devices to be integrated into the system easily. Thirdly, the integration of learning from acquired knowledge can be used for demand dispatch and other planning purposes. Moreover, the learning capability of flexometer can upgrade smart appliances to be more intelligent.

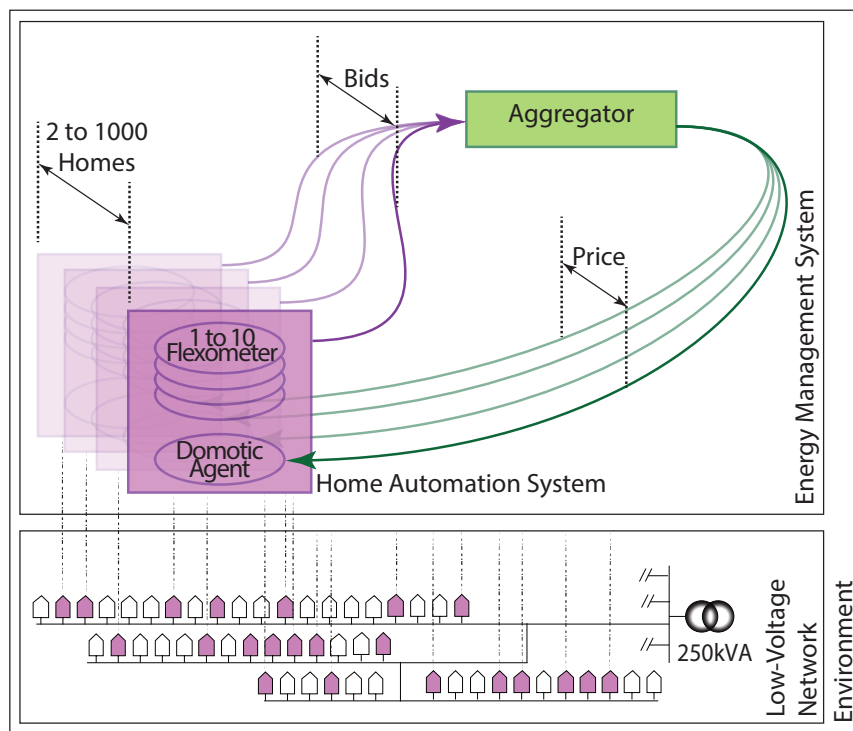


Figure 1. Multi-agent based energy management system using transactive energy.

2.2. Design

In order to implement Energy Flexometer in EMS, the authors proposed the Echelon's IzoT platform. The IzoT is considered to be a new version of the LonWorks technology dedicated for BACS, using IP-all-the-way connections to the end devices. It has already provided a ready to develop platform (including microprocessors, application programming interfaces, communication protocol, management and integration software). According to the LonWorks standard, an interoperability between IzoT nodes is provided by functional profiles. Based on this platform, a standardized design of the Energy Flexometer is proposed in this section. Energy Flexometer supports three main functions: (1) an Energy Meter, (2) an Energy Logger and (3) an Elasticity Learner, as shown in Figure 2.

Functional profiles provide definitions both for network variables (NVs) as well as configuration properties (CPs), which are included in functional blocks as per algorithm requirement. Moreover, functional blocks proposed in this paper are designed according to the Semantic Device Descriptions model presented in [37]. This way they are open and ready to use in Component-based Automation Systems model introduced in [38].

All NVs and algorithms proposed in this paper are universal and could be seamlessly integrated in other international BACS standards networks. Thus, interoperability is one of the most important objective of the Energy Meter functional profile. The developed profile describes the application

layer interface (*NVs*, *CPs*) and defines functional blocks proposed for this application. The *NVs* are essential elements of *BACS* module's network interface for binding network variables from other nodes. They are defined with prefix of *nvi* for inputs and *nvo* for outputs, providing data and information in the *BACS*, simplifying the integration process (development and installation of distributed systems). Moreover, according to the LonWorks standard assumptions and requirements, all *NVs* are optimized as short data objects to minimize load of the data communication channels. It is important taking into account further implementations of the proposed concept in larger *EMS* systems. In this way the *BACS* devices can be defined individually, then easily rearranged into new applications. The *NVs* are essential for interoperability between nodes. Herein, the paper designs the functional blocks in such a fashion that they are collectively able to express all kind of primary process parameters and customer preferences. Bearing in mind all these technological aspects, the proposed Energy Flexometer concept is ready to implement in different applications, both small and large *EMS* systems. However, the scalability is not considered in this paper as it will be ensured due to well known, standardized and proof in many applications LonWorks technology and other open, international *BACS* standards [39,40].

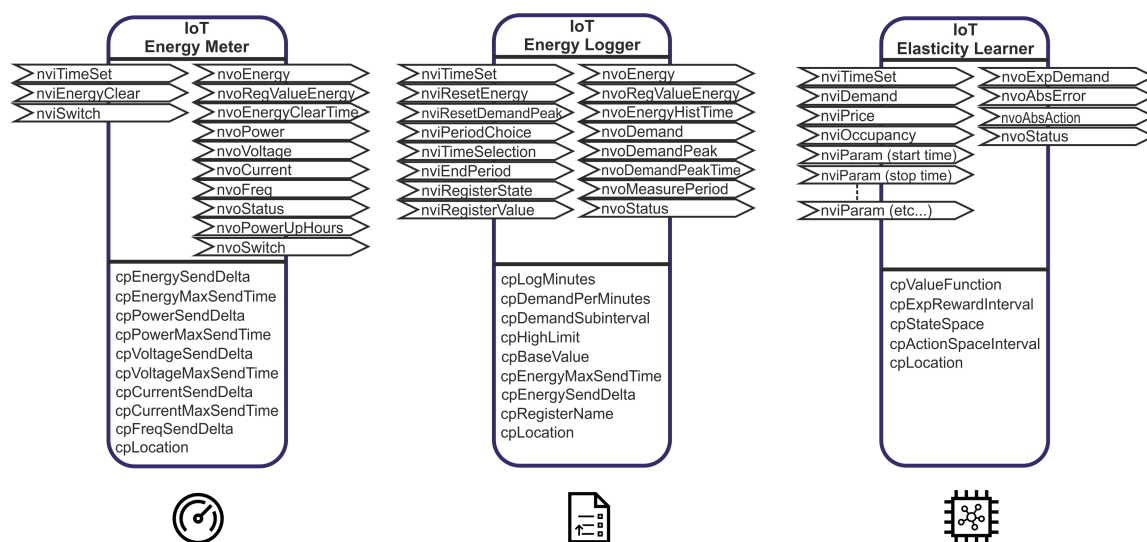


Figure 2. Developed Energy Flexometer functional blocks.

2.3. Energy Meter

Changes within the device and its primary process parameters are acquired by energy meter on real-time basis. The energy meter also captures changes that may occur due to customer preferences (e.g., user implicitly controlled or *EMS* explicitly controlled the device) or may occur naturally due to environmental change (e.g., room temperature). Then energy meter stores the current primary process parameters and customer preferences as a table with columns for all network variable and configuration properties respectively.

Table 1 shows the most important *NVs* in *IoT* Energy Meter functional block, providing electrical parameters measurements, such as *nvo*Energy, *nvo*Voltage, *nvo*Current, *nvo*Power or *nvo*Freq. Mentioned *NVs* are meter value output, i.e., the actual running value with timestamps. All these output *NVs* are transmitted when polled, or are triggered by Send On Delta condition—for communication settings adjustment *cp*ParamSendDelta and *cp*ParamMaxSendTime are proposed (where Param could be Energy, Power, Voltage, Current or Freq). The *nvo*Energy could be set to zero using *nvi*EnergyClear, nevertheless total active energy value is non-resettable and provided by *nvo*ReqValueEnergy. Controls on the load or group of loads connected to the Energy Flexometer is provided by the *nvi*Switch and *nvo*Switch, i.e., they are dedicated for controlling the state of relay actuator. Moreover, *nvo*Status variable contains the data related to the internal status conditions of the energy meter.

Table 1. IoT Energy Meter.

NV Name	Description
nviTimeSet	Synchronization and time settings for measurement purpose
nviEnergyClear	Reset or initialize the electricity consumption total
nviSwitch	Load control input—to provide external signals (e.g., an request for energy reduction sent by the provider to the customer in DR services)
nvoRegValueEnergy	Total active energy—meter value output
nvoEnergyClearTime	Total active energy—this NV cannot be reset
nvoPower	Date and time of resetting nvoEnergy
nvoVoltage	Total active energy—meter value output
nvoCurrent	Phase voltage mean value
nvoFreq	Phase current mean value
nvoStatus	Fundamental voltage frequency
nvoPowerUpHours	Device status report/info
nvoSwitch	Operating hours since the last time operating at which the device was switched ON
cpParamSendDelta	Load control output—providing direct control of device or group of loads, taking into account demand analysis results, by changing the state of the relay actuator
cpParamMaxSendTime	Send on Delta send condition setting (Param can be energy, current, etc.)
cpLocation	Polling time send condition setting (Param can be energy, current, etc.)
	Installation location and logger ID

2.4. Energy Logger

Once the energy meter has acquired agent state (i.e., primary process parameters and customer preferences), energy logger then separates it into events. Each event denotes an instance with respect to the state and a control action that was being performed. Then the pair of action and state is logged into the logger.

To apply learning algorithm, the state must be mapped to a Markov decision process (MDP) consisting of a data tuple (state, action, transition probability, reward). State x_k of an agent includes all possible network variables and configuration properties. Action u_k captures the control action to an agent (e.g., turn on/off lamp). Transitional probability is a vector describing the transition of a agent from current state to a new state for a given action. Reward r_k herein is simply a marginal energy cost incurred by an agent for the given state x_k and action u_k . The general rule in a competitive market environment is that the profit can be maximised at the quantity of output where marginal revenue equals marginal cost. Hence, r_k in the given formulation can be defined as follows:

$$\lambda_k = \Gamma_k \times \left(1 + \frac{1}{\epsilon_k}\right) \quad (1)$$

where $\Lambda \doteq [\lambda_1, \dots, \lambda_K]$ represents the vector of marginal cost which is expected to incurred by Flexometer. Γ_k represents market price during k^{th} interval. ϵ_k is a price elasticity of demand.

The IoT Energy Logger functional block include NVs and CPs related to its functions, as shown in Table 2. Essential for the Energy Logger are NVs providing information about energy. The first one nvoEnergy is a copy of the current meter value (for the last month). With this network variable, it is also available to display historical data stored by the unit. The desired output data could be selected by the nviTimeSelection variable. After setting nviTimeSelection, the nvoEnergy is updated with the data of the cumulative meter value as it was for the requested time. The nviTimeSelection controls, according to the time, which history value is shown on the output network variable side via the nvoEnergy and nvoEnergyHistTime. If the time is outside of the accepted range the register output nvoEnergy is zero and the status field indicates “Illegal value request” information. Furthermore, a group of NV dedicated to power demand handling is important as well. For example, the nvoDemand holds

the demand value. It is related to average power calculated for a specified time interval. Moreover, a rolling demand function with “sliding window” mode is supported by the IoT Energy Logger as well. In this case, demand calculation is carried out for a fixed number of subintervals, providing average power value for specific time interval. This results in better accuracy, especially for demand peaks (nvoDemandPeak and nvoDemandPeakTime). The nvoStatus provides information about activated algorithms for demand calculation as well as modes for controls and operations.

Table 2. IoT Energy Logger.

NV Name	Description
nviTimeSet	Synchronization and time settings for logging purpose
nviResetEnergy	Reset or initialize the nvoEnergy
nviResetDemandPeak	Reset Demand Peak NVs
nviPeriodChoice	Historical period of when data are saved to the meter
nviTimeSelection	Historical time selection
nviEndPeriod	Measuring period ending input
nviRegisterState	Register state selection input
nviRegisterValue	Send value to the energy register object via the network
nvoEnergy	Total active energy—copy of the current meter value (for the last month). It could provide other historical data as well
nvoRegValueEnergy	Current value of the energy register with a time stamp and status bits
nvoEnergyHistTime	Register historical time output
nvoDemand	Demand power (average power over demand period)
nvoDemandPeak	Peak demand power
nvoDemandPeakTime	Time and date of peak demand
nvoMeasurePeriod	The length of the measuring period
nvoStatus	Device status report/info
cpLogMinutes	Logging interval. Default: 15 min
cpDemandPerMinutes	Demand period: 5 min to 1440 min. Default: 15 min
cpDemandSubinterval	Rolling demand subinterval count: 1–8. Default: 1
cpHighLimit	Definition of the highest normal value of the register
cpBaseValue	Definition of the base value of the register
cpEnergyMaxSendTime	Polling time send condition setting (Param can be energy, current, etc.)
cpEnergySendDelta	Send on Delta send condition setting (Param can be energy, current, etc.)
cpRegisterName	Definition of the register name
cpLocation	Installation location and logger ID

2.5. Energy Elasticity Learner

This functional block has an objective to forecast a state of an agent for an expected action taking into account the price elasticity of demand. It has been found in [13] that price elasticity of demand can be successfully use for estimating agent’s value-proposition (i.e., bid). Demand elasticity is defined as the change in demand $\partial x_{d,a}^k$ at an interval k due to the change in the price $\partial \Gamma^k$ during the same interval. Mathematically,

$$\varepsilon_{kk} = \frac{\Gamma^0}{x_{d,a}^0} \frac{\partial x_{d,a}^k}{\partial \Gamma^k} \quad (2)$$

where ε_{kk} is the elasticity coefficient that indicates demand flexibility, $x_{d,a}^k$ is flexible demand during time interval k , Γ^k is price signal during the same interval yielded by the domotic agent, $x_{d,a}^0$ is initial flexible demand and Γ^0 is the initial price signal. From an economic point of view, ε_{kk} represents the self-elasticity and can be used to calculate demand sensitivity of an appliance concerning the price signal. However, the concept (as shown in (2)) cannot be generalized for the TE because it is imprecise to use initial or reference states for elasticity calculation at the access layer in a real-time environment. In this regard, the precise calculation of the demand elasticity at the access layer can be performed by using the concept of arc elasticity, defined by Seldon in [41]. Arc elasticity calculates the change in

percentage relative to the mid-points, thus resulting in (a) symmetrical change concerning the price and demand; (b) relational independency and (c) unity provided the total revenue at both points is comparable. Mathematically,

$$\varepsilon_{kk} = \frac{\frac{\Gamma^k + \Gamma^0}{2}}{\frac{x_{d,a}^k + x_{d,a}^0}{2}} \frac{\partial x_{d,a}^k}{\partial \Gamma^k} \simeq \frac{\Gamma^k}{x_{d,a}^k} \frac{\partial x_{d,a}^k}{\partial \Gamma^k} \mid \Gamma^k \simeq \Gamma^0, x_{d,a}^k \simeq x_{d,a}^0 \quad (3)$$

The composite bidding rules, as mentioned in [42], so according to (2) the demand elasticity of the current k^{th} interval versus an interval $k' : k' \neq k, \forall k' \in \mathcal{K}$ can be defined as:

$$\varepsilon_{kk'} = \frac{\Gamma^{k'}}{x_{d,a}^k} \frac{\partial x_{d,a}^k}{\partial \Gamma^{k'}} \quad (4)$$

$\varepsilon_{kk'}$ corresponds to the cross elasticity. Hence, by combining self-elasticity and cross-elasticity in a matrix results in demand elasticity matrix. Mathematically, it can be shown as:

$$\epsilon = \begin{bmatrix} \epsilon_{(p,p)} & \epsilon_{(f,p)} \\ \epsilon_{(p,f)} & \epsilon_{(f,f)} \end{bmatrix} = \left[\begin{array}{cc|cc} \ddots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \varepsilon_{(-2,-2)} & \varepsilon_{(-2,-1)} & \varepsilon_{(-2,0)} & \varepsilon_{(-2,1)} & \cdots \\ \cdots & \varepsilon_{(-1,-2)} & \varepsilon_{(-1,-1)} & \varepsilon_{(-1,0)} & \varepsilon_{(-1,1)} & \cdots \\ \cdots & \varepsilon_{(0,-2)} & \varepsilon_{(0,-1)} & \varepsilon_{(0,0)} & \varepsilon_{(0,1)} & \cdots \\ \cdots & \varepsilon_{(1,-2)} & \varepsilon_{(1,-1)} & \varepsilon_{(1,0)} & \varepsilon_{(1,1)} & \cdots \\ & \vdots & \vdots & \vdots & \vdots & \ddots \end{array} \right] \quad (5)$$

where $\epsilon_{p,f}$, referred to a postponing cross-elasticity, maps the past input to the future output, thus receiving all necessary information about the previous behaviours/states of the consumer/agent that may influence the expected states. $\epsilon_{f,p}$, referred to as advancing cross-elasticity, maps the expected future input to past output, thus receiving the prediction of future behaviour of an agent.

Table 3 shows the Energy Elasticity Learner functional block's NVs and CPs. In this case, crucial network variable is nvoExpDemand related to expected demand value. It provides an information from online learning demand process taking into account additional input parameters. Moreover, the input nviDemand enable to get information about current demand value provided by the nvoDemand—from the IoT Energy Logger functional block. Both mentioned NVs (nvoExpDemand and nviDemand) could be correlated and compared with actual demand value. In this way they provide information holding in nvoAbsError. It is important for management and control of loads in the EMS and it could be used by the Domotic Agent module described in the FPAI concept.

The nviPrice provides information about changes in an electrical energy price, taking into account external signals. For example, information about a higher energy price effects on changes in load profile for the building/object and could initiate load control and shifting process. The nviOccupancy represents information about presence of persons in rooms and affects calculated value of nvoExpDemand depending on occupancy. The input nviParam is an action for the learning process to calculate nvoExpDemand and it is associated with customer preferences. Every nviParam is updated by a customer before the activation of an agent. The most common temporal preferences could be start time and stop time to the agent, which means during a day the agent can start from the given time and must complete its task before the identified time. In this way, it can be inferred that nviParam has a direct influence on PEM. The nvoStatus provides information about selected demand elasticity calculation and learning algorithm as well as control and operating modes. This information could be used by other BACS devices and energy provider, operating mode of the DR services to the Domotic Agent and energy provider. The CPs allow to adjust settings for proper work of the IoT Elasticity Learner.

Table 3. IoT Energy Learner.

NV Name	Description
nviTimeSet	Synchronization and time settings for learning purpose
nviDemand	Demand value input from IoT Energy Logger
nviPrice	Customer preferences parameter which may affect demand—Price
nviOccupancy	Customer preferences parameter which may affect demand—Occupancy
nviParam (start time)	Agent start time
nviParam (stop time)	Agent stop time
nviParam (etc...)	customer preferences input parameter which may affect demand
nvoExpDemand	Expected demand value (calculated)
nvoAbsError	Absolute error value (of calculation)
nvoAbsAction	Action value (of calculation)—control action to an agent (e.g., turn on/off lamp)
nvoStatus	Device status report/info
cpValueFunction	Setting: learner value function
cpExpRewardInterval	Send condition: maximum reward
cpStateSpace	Setting: triggerings actions
cpActionSpaceInterval	Send condition: nvoExpDemand
cpLocation	Installation location and logger ID

3. Demand Elasticity Estimation

Although there are many variants of machine learning, this study considers Q -learning technique for solving the problem because it is used to learn primitive behaviours of an agent. Q -learning is a machine learning proposed in [43] for determining the Markov decision process with fragmentary knowledge. Q -learning is about the training of animal's behaviour in an environment. That is why, social cognitive theory considers the concept of Q -learning for mimicking social responses. Therefore, demand elasticity estimation holds Q -learning to emulate the consumer's demand flexibility. Q -learning technique consists of an agent that takes actions within an environment and receives respective experiences in the form of rewards for all possible states. The technique is suitable for evaluation of demand elasticity because (a) it can form the problem in (4) as a combinatorial optimisation problem; (b) the consumer preferences (e.g., NVs and CPs) can help in perceiving the environment; (c) Q -value simulates the consumer behavior easily and (d) the continuous updating process results in better estimation.

Therefore, this section explains in detail action space, state space, a reward function and action selection for the estimation of demand elasticity.

3.1. State Space and Action Space

The state carries the necessary knowledge (i.e., the price of electricity) for making a decision. So, during each interval $k \in K$, the state $\{s_a^1, s_a^2, \dots, s_a^K\}$ represents the day-ahead price to the elasticity agent.

However, the action (δ_a^1) represents price expectation (Γ^k) for a given state. Within the given state, an agent selects an action $\{\delta_a^1, \delta_a^2, \dots, \delta_a^k, \dots, \delta_a^K\} \in \hat{\delta}_a$. Consequently, the agent receives an estimated reward $r_{i+1} \in \mathcal{R}$ and a new state s_a^{i+1} , where i is iteration number.

3.2. The Objective Function

The objective of the proposed problem is to estimate the demand elasticity (as discussed in (4)) that can be inferred from the finding of a suitable sequence of actions $\{\delta_a^1, \delta_a^2, \dots, \delta_a^k, \dots, \delta_a^K\} \Rightarrow \{\Gamma^1, \Gamma^2, \dots, \Gamma^k, \dots, \Gamma^K\}$. Inference can only be legitimate provided $\epsilon_{f,p}$ should be maximized and should keep the total revenue positive.

For an objective function, presume $r^i(s_a^i, \delta_a, s_a^{i+1})$ be an transitional elasticity between s_a^i and s_a^{i+1} . Remember, the revenue or profit maximisation does not always result in an optimal strategy if the objective is to explore the total available flexibility of demand. Consequently, the maximisation of r^i would explore the total flexibility and would generate a bid that maintains an equilibrium such that the total marginal revenue returns in the least.

3.3. Action Selection

In order to understand the selection of suitable action, let s^i be the current state at the interval k . During the i^{th} iteration, an action δ^i is being randomly selected through an ϵ -greedy algorithm. Consequently, it receives r^i . However, to achieve an optimal action δ_a^* , the agent looks into the Q -Value table. The iterative process may initially selected action δ_a^i , thus δ_a^* can be seen as an odd recursion at first because it is expressing the Q -Value of an action in the current state regarding the best Q -Value of a successor state. However, δ_a^i makes sense when you look at how the exploration process uses it. The process stops when it reaches a goal state (i.e., s^i) and collects the reward (i.e., r^i), which becomes that final transition's Q -Value. Now in a subsequent training episode, when the exploration process reaches that predecessor state, the method uses the above equality to update the current Q value of the predecessor state. So, if $Q(s_a, \delta_a^1), Q(s_a, \delta_a^2), \dots, Q(s_a, \delta_a^m), \dots, Q(s_a, \delta_a^M)$ are the Q -values against respective actions, then δ_a^* is an optimal action provided $Q(s_a, \delta_a^*) > Q(s_a, \delta_a^i)$. Mathematically,

$$\delta_a^* = \arg \max_{\delta_a^i \in \hat{\delta}_a} Q(s_a, \delta_a^i) \quad (6)$$

Next time its predecessor is visited (i.e., s_a^{i+1}) that state's Q value gets updated, and so on back down the line. Provided every state is visited infinitely often this process eventually computes the optimal Q .

$$Q^{i+1}(s_a, \delta_a) = (1 - \alpha)Q^i(s_a, \delta_a) + \alpha[r^i + \max_{\delta_a^i} Q^i(s_a^{i+1}, \delta_a^i)] \quad (7)$$

where $\alpha \in (0, 1]$ represents learning coefficient. Remember, in the initial learning process, the optimal action may not be the best action. Therefore, the goodness of the algorithm depends on the balance between the exploitation and exploration of the previously acquired knowledge. The ϵ -greedy based Q -learning, which is used herein, is incredibly simple and often maintains the right balance between exploitation as well as exploration, as shown in Algorithm 1. Moreover, the algorithm fits well within the domain of the proposed problem, i.e., learning of demand elasticity. As a flexometer plays the algorithm, it keeps track of the average marginal cost of an appliance. Then, it selects the state of appliance with the highest current average marginal cost (i.e., r) with probability $= (1 - \epsilon) + (\epsilon/k)$, where ϵ is a small value like 0.10. In addition, it selects states that do not have the highest current average marginal cost with probability $= \epsilon/k$.

Algorithm 1: Demand Elasticity Estimation Algorithm.

```

input :bid  $\eta_{d,a}$ 
input :price signals ( $\Gamma$ )
1 initialization;
2 forall for all time intervals  $k \in \mathcal{K}$  do
3   repeat
4     initialization  $\hat{s}_a^i \leftarrow \hat{s}_a^1$ ;
5     repeat
6       Choose  $\delta_a^i$  from  $\epsilon$ -greedy algorithm;
7       Select  $\hat{s}_a^{i+1}$  from  $\{s_a^1, s_a^2, \dots, s_a^K\}$ ;
8       Observe  $\lambda_{d,a}^k$  by using (1);
9       Find  $Q(\hat{s}_a^i, \delta_a^i)$  by using (7);
10       $\hat{s}_a^i \leftarrow \hat{s}_a^{i+1}$ ;
11     until  $\hat{s}_a^i \leftarrow \hat{s}_a^1$ ;
12     update:  $\epsilon$  and  $Q(\hat{s}_a^i, \delta_a^i)$ 
13   until for each iteration  $i$ ;
14   update:  $\mathcal{P}_a^k$ 
15 end

```

3.4. Simulation of Demand Elasticity by using the Algorithm

In order to analyse the efficacy and performance of the proposed algorithm is analysed, a case is studied herein. In this case, it is assumed that there are day-ahead prices for around 56 days with a granularity of 15 min (i.e., 56 days \times 96 intervals, implies to = 5376 total intervals) However, an action space $\mathcal{A}(s_k) = \{0 : 100\}$ represented as indexes to price levels. Moreover, the vector of best actions $\hat{\delta}_a = \{\delta_a^*(s_1), \delta_a^*(s_2), \dots, \delta_a^*(s_k), \dots, \delta_a^*(s_K)\}$ is logged after every 96 intervals. As the vector of best actions represents the indexes to day-ahead prices, so the vector of expected day-ahead price can be easily generated. Furthermore, through out the simulation after 96 intervals, (8) is used to update Probability Density Function (PDF). PDF support in retrieving expected day-ahead prices. For an explanation, the most updated PDF, (i.e., \mathcal{P}_a^k), obtained at the end of simulation is presented in Figure 3. It can be observed that the vector of best actions has learned the price transitions because the line (representing the day-ahead prices) in the figure matches the PDF, which was updated from the vector of best actions.

$$\mathcal{P}_a^k(\delta_a^i) = \begin{cases} \mathcal{P}_a^k(\delta_a^i) + \eta(1 + \mathcal{P}_a^k(\delta_a^i)) & , \text{if } \delta_a^i = \delta_a^* \\ \mathcal{P}_a^k(\delta_a^i) - \eta\mathcal{P}_a^k(\delta_a^i) & , \text{otherwise} \end{cases} \quad (8)$$

In order to further study the effectiveness and performance of a learning process, an absolute error which calculates the differences between the expected day-ahead price and the real price signal is shown in Figure 4. The decreasing trend in an error provides an evidence that the proposed learning technique works seamlessly.

On the other hand, in order to study the accuracy of demand elasticity estimation, a numerical calculation for the entire duration of the simulation is performed by using (4). Then the numerically calculated demand elasticity is compared with estimated demand elasticity. In this regards, Figure 5 shows the comparison between the two different elastic values. Similarly, the diminishing of trend in comparison error is in line with the results shown in Figure 4. Means, the initial expectations were less accurate, but once the agent was learned then it results in less error. However, an interesting observation is Figure 5 is that the elasticity is less predictable in the last intervals for each time window (i.e., 96 intervals). The reason was related to flexibility allocation because all the flexible loads must be utilized at the end of the day that results in inelastic demand. Moreover, the effect was very prominent because in this study it assumed the domestic agent does not have scheduling capability. For the same reason, the unexpected high errors, that are in Figure 4, can happen even after learning.

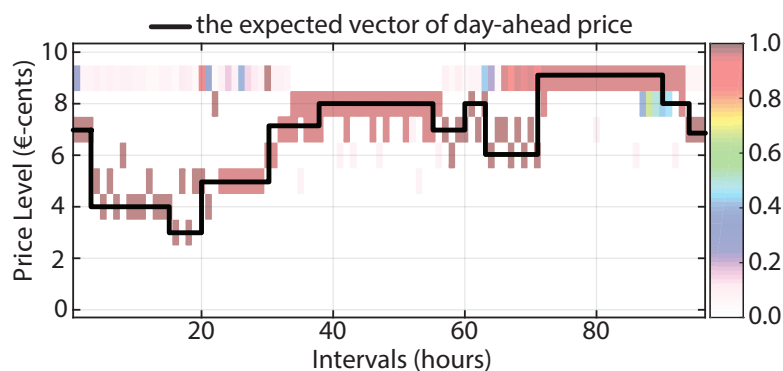


Figure 3. The most updated PDF, (i.e., \mathcal{P}_a^k), obtained at the end of the simulation.

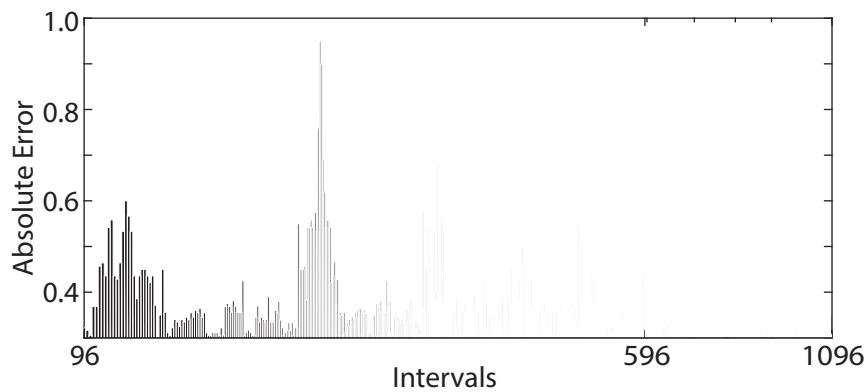


Figure 4. The absolute error of the expected and the real price.

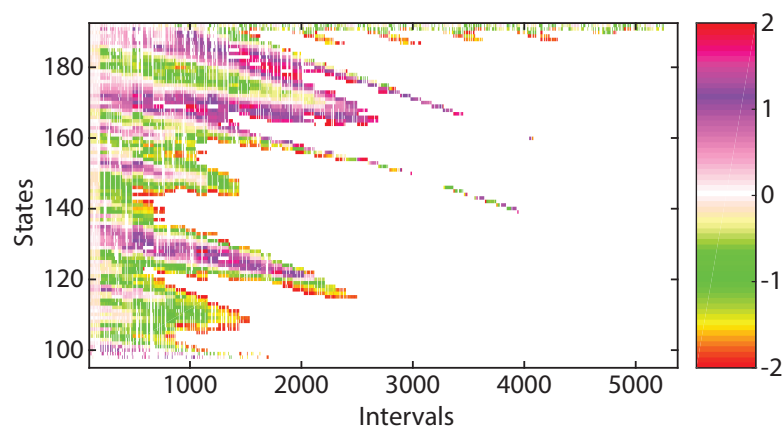


Figure 5. The difference between the numerically calculated elasticity and the elasticity estimated by the proposed algorithm.

Hence, the main conclusions from the results are two-fold. Firstly, the approach would learn the consumer behaviour in a number of intervals even-though there was not much information provided initially. Secondly, it would help the domotic agent to build sensitive and agile demand response programs for demand scheduling in day-ahead as well as real-time.

4. Physical Implementation

The functional blocks proposed and described in Section 2 were implemented in IzoT intelligent node as a logical interface of the Energy Flexometer. The IzoT device stack was based on Raspberry Pi 2 Model B Boards, with 900 MHz quadcore ARM processors and 1 GB of memory with an additional integrated power measurement circuit. This platform was chosen because of its versatility and ease of implementation. It provides an extensive environment for application development and tests.

4.1. Measurement System Design

The measurement system was developed with two CS5460 analog to digital converters. It was dedicated to measure essential electrical parameters such as: Real Energy, RMS voltage, RMS current, and Instantaneous Power for single phase 2- or 3-wire installations. In the proposed system, the CS5460 was connected with Raspberry Pi microcontroller using general purpose in/output pins. System structure and all the connections are presented in Figure 6. Further detail information about the measurement system design are presented by authors in [15].

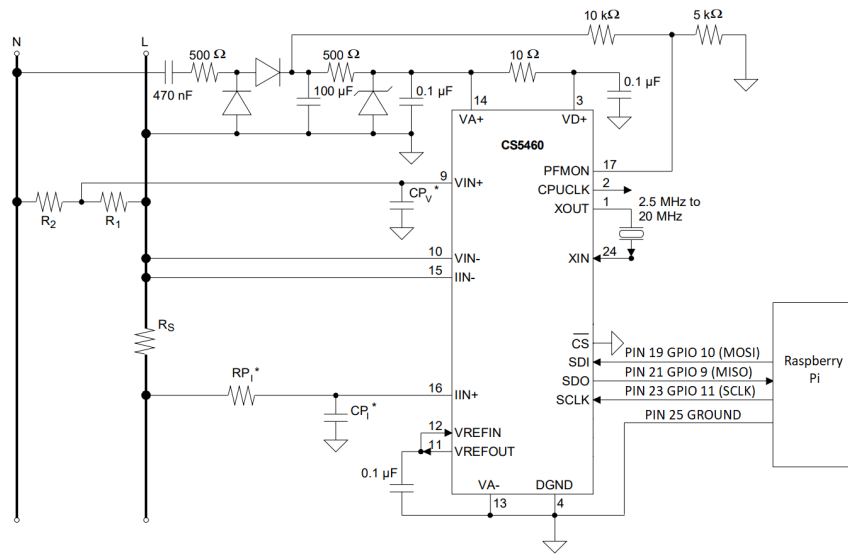


Figure 6. A schematic diagram of Flexometer with CS5460 IC and Raspberry Pi microcontroller.

4.2. Finite State Machine

In the phase of logical implementation, an application for the Flexometer has been developed. It was implemented in Raspberry Pi using the IzoT stack and provided: (1) serial communication interface for Raspberry Pi and CS5460, (2) reading of registered values from the CS5460, and (3) an application code for control and learning was written in C programming language [14,15]. Figure 7 shows finite state machine of the Flexometer, which has 7 states (i.e., configuration parameters and control) and 7 transitions (i.e., network variables). This provides a detail description of an agent in terms of a digital logic circuit function at a given instant in time, to which the state circuit or program has access.

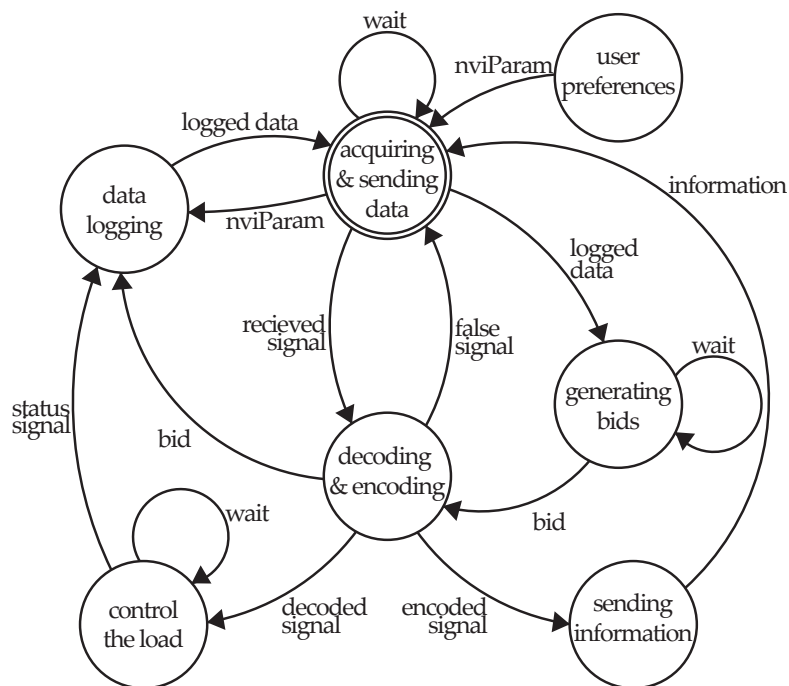


Figure 7. Finite state machine of the Energy Flexometer.

4.3. Knowledge Base

The knowledge based ontology of the Flexometer, represents the functional relationship of all blocks within the Flexometer, is shown in Figure 8. It contains descriptions of functional blocks and their actions, as well as reference for calculating state change when a control action (i.e., price signal) is given. Basic customer preferences (i.e., nviParam like start and stop time) are also contained in this ontology. As can be seen in Figure 8, the logger functional block is associated with customer preference, demand response and learned value proposition.

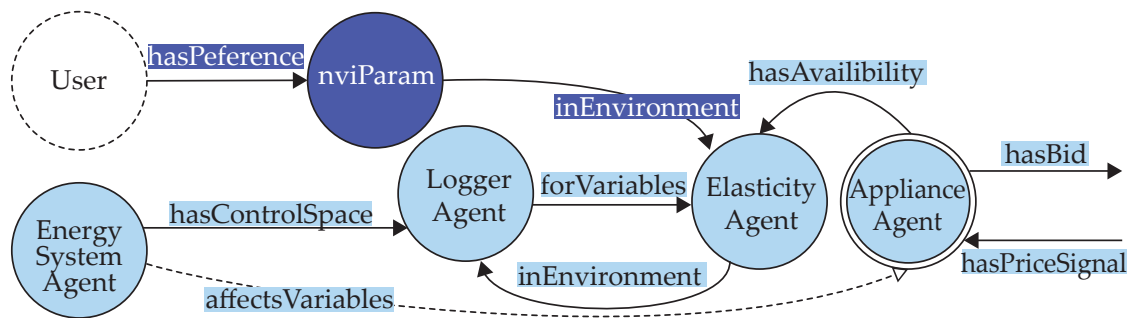


Figure 8. Knowledge based ontology used for representing functional blocks within the Flexometer and their relationships.

5. Evaluation

In this paper, a hardware-in-loop approach was adapted to conduct an experimentation for the evaluation of the Flexometer as an element that supports in the integration of MCM in EMS.

5.1. Experiment Design

As shown in Figure 9, the system for the experimentation was designed by implementing multi-agents. In this architectural design, multiple agents were responsible to perform their respective tasks. Herein, agents were also organized in triple layers. Agent in most upper layer was called an aggregator agent. The aggregator agent aggregated bids received from domestic agents and then adjusted an equilibrium price signal as per the objective. The most simple objective of the aggregator is to balance supply and demand, which was taken under consideration in this study. As shown in Figure 9, herein aggregator agent simply broadcasted the price signal corresponding to nearly zero consumption as the equilibrium price signal.

On the other hand, domestic agents exist in the middle layer of the organization. Herein, domestic agent worked as a transceiver of bid and price signal between the connected appliance agents and the aggregators. In this architectural design, appliance agent were representatives of real physical load (like battery, PV system or other loads) to the domestic agent. Therefore, within this framework, “Flexometer” was an appliance agent with digital logic circuit. Therein it provided an opportunity for the standardized integration of physical load into Transactive-based control mechanism.

As mentioned, the purpose of this demonstration was to evaluate the Flexometer. So, the demonstration was planned for a time period of a week in the Smart Lab of AGH UST, Krakow–Poland. Moreover, in order to simplify the analysis of data obtained during demonstration, the granularity of an hour was considered. Figure 10 shows the illustration of lab setup for experimentation. It can be observed from Figure 10 that an aggregator agent was entirely developed in MATLAB run-time environment. In this experimentation, two domestic agents were designed, one was MATLAB-based and other was designed in Raspberry Pi. Python and C++ languages were used to implement the logic of domestic agent in Raspberry Pi.

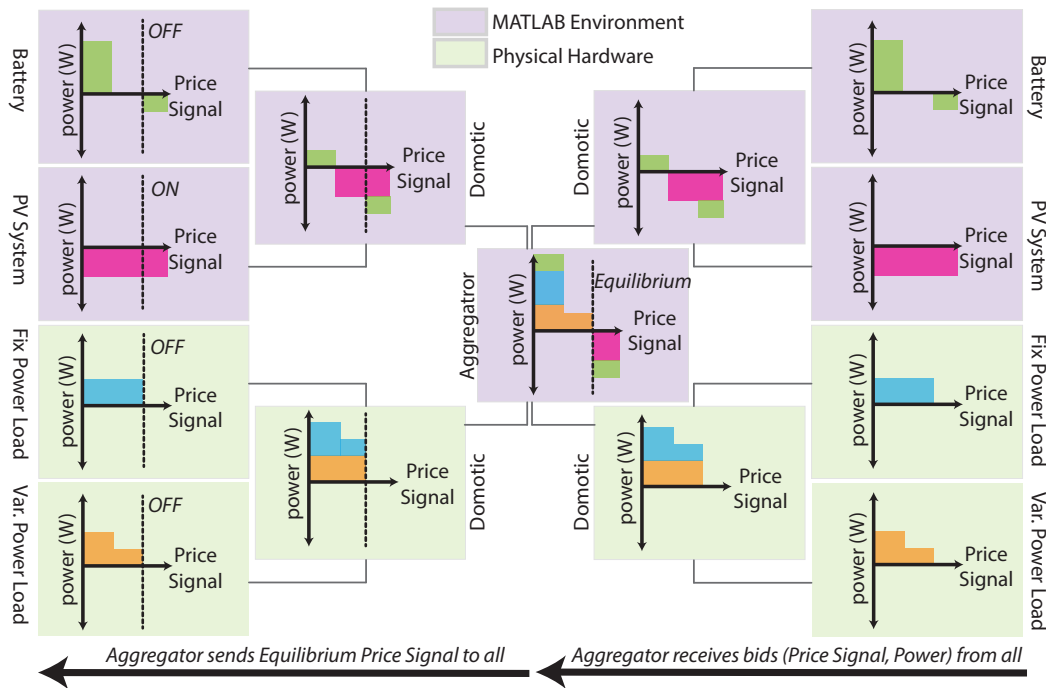


Figure 9. Three layer multi-agent system for Transactive-based control mechanism.

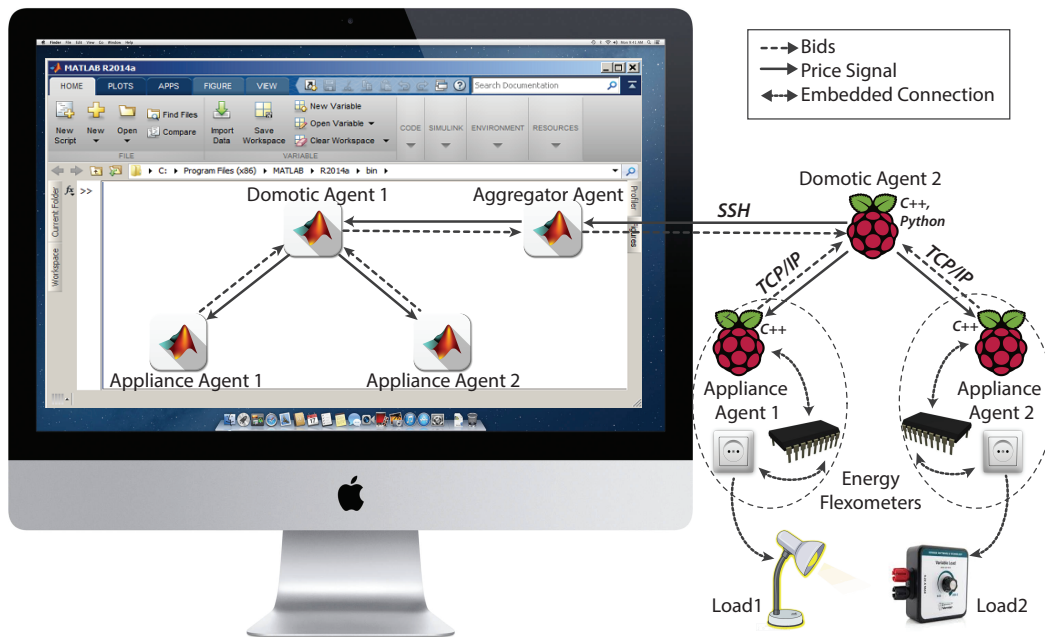


Figure 10. Setup of multi-agent system in Lab.

Moreover, each domotic agent was equipped with two appliance agents. Appliance agents, which were connected to MATLAB-based domotic agent, were also modeled in MATLAB, as shown in Figure 10. Out of two MATLAB-based appliance agents; one was modeled as a battery of maximum rated absolute power of 3 kW and other was modeled as a PV system of maximum peak power of 2.6 kW. For PV system, to have more real life experience, local irradiation values of a week in June were considered because during the period PV generates the maximum peak concerning the entire year. On the other hand, two Flexometers were implemented as per the design presented in Figure 4. Out of two Flexometers, one was connected to fixed power dummy load of 2 kW and other was connected

to variable power dummy load (i.e., [min, max] = [0.5 kW, 4 kW]). Variable power dummy load was used to generate variable power profile for a defined time duration (i.e., maximum upto 7 h). Figure 11 shows box diagram of the power pattern by variable load for the time period of a week. It also shows the average consumption pattern (by a red line in the figure) of the variable load with respect to the time.

Moreover, as two temporal preferences (i.e., nviParam) were introduced as customer preferences, namely start time and stop time. Start time is a time that refers to an hour of a day from when device can be turn on, however stop time is a time that refers an hour of a day till when device must complete its defined task. The mean stop time and start time for fixed load were from 4th hour till 19th hour of the day. On the other hand, the mean stop time and start variable for fixed load was from 2nd hour till 20th hour of the day. Keeping in mind that the day of the demonstration starts at 9.00 a.m.

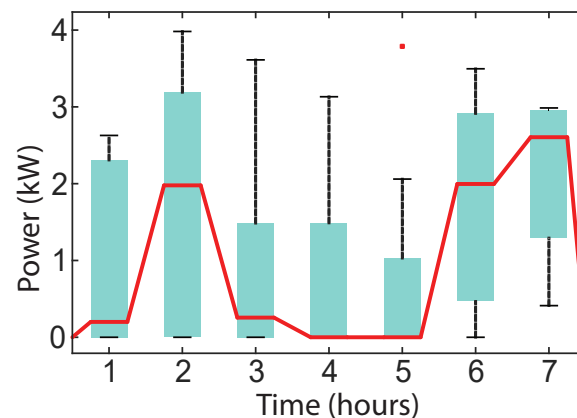


Figure 11. Average consumption pattern (in red line) and box plot of variable load for a time period of a week.

5.2. Performance Metrics

The performance metrics of Flexometer are training time taken by elasticity learner functional block and response time. When numerically calculating without nviParam for a learning task of elasticity learner programmed in Raspberry Pi 2, the algorithm that runs for 100 episodes should take 8.3 s on average. However, Table 4 shows average training time that the elasticity learner of each Flexometer takes in both situations i.e., with nviParams and without nviParams. It can be observed that time taken with nviParam is way lower than without nviParam. It provides an evidence to the fact that nviParam limits the exploration across state space during a training episode, thus allowing agent to converge faster. Moreover, the average response time, i.e., a time duration required to change the status of Flexometer, was found to be 5 ms.

Table 4. Average training timing with and without nviParam.

Flexometer	Without	With
with variable loading	13 s	7.0 s
with fixed loading	4.9 s	1.7 s

Observation 1. *If an appliance agent conforms to the complex bidding rules [2], then the computation time required for the learning process will be lower than the simple bid. Because complexities in the bid limit the exploration across the state space during a training episode, thus allowing the agent to converge faster.*

5.3. Results

The upper graphs in Figures 12 and 13 show bid (i.e., power versus price signal) generated by both Flexometer (variable and fix load) during two different demonstrations i.e., with or with nviParam respectively. Similarly, the lower graphs in Figures 12 and 13 show when both Flexometer turn ON verses the respective price signal they received from domotic agent for an action. From Figure 12, an evident observation is that Flexometers without nviParam turn ON more strictly on time as well as react on high values of price signal. On the other hand, as shown in Figure 13, Flexometers with nviParam turn ON to relatively lower values of price signal as well as dispersed more on time. This provides an additional fact that elasticity learner learns more strict bidding in case of without nviParam rather than with nviParam.

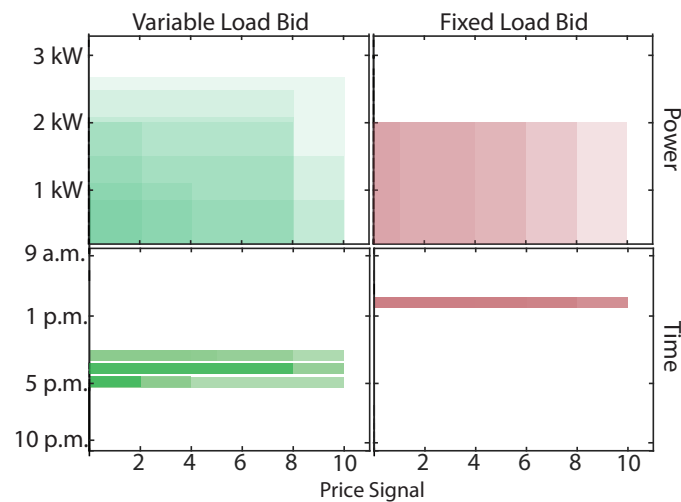


Figure 12. Bid generated without nviParam by Flexometers and their turn ON timings during demonstration with respect to price signals.

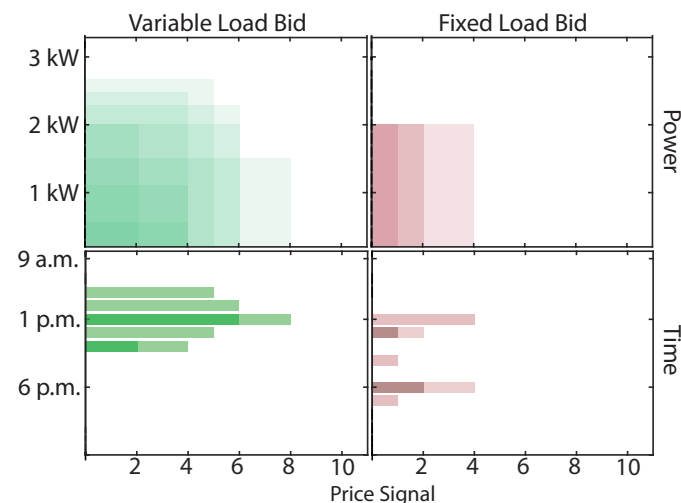


Figure 13. Bid generated with nviParam by Flexometers and their turn ON timings during demonstration with respect to price signals.

Observation 2. *If an appliance agent conforms to the bidding rules, then it turns ON to relatively lower values of the price-signal as well as relatively dispersed more throughout the day. Moreover, the elasticity agent learns relatively strict demand elasticity with respect to time in case of the simple bid rather than the complex.*

6. Conclusions

In this paper, a new solution approach to organizing active connection between the local EMS and market-driven DR mechanism, within the TE framework has been introduced, taking into account the IoT technology as a universal data communication platform. An essential element of the proposed solution is an Energy Flexometer, which is a key part of any EMS and in this paper it is considered to be an agent-node of the proposed MAS. Moreover, in this research the authors developed new functional blocks for the Energy Flexometer, designed according to the LonWorks standard, implementing the IoT technology. They are considered as a logical interface of the Energy Flexometer and provide standard network variables and configuration properties to organize advanced and fully integrated control and monitoring systems within both field level as well as IP communication networks. This approach allows using the designed and developed EMS platform with Energy Flexometer in the TE framework. Abovementioned comprehensive solution allows to implement the Energy Flexometer concept both in new EMS applications as well as existing ones, without any additional requirements. Due to the standardization of logic interface, proposed by authors, fully interoperability is provided. Moreover, the scalability is ensured and determined by using IzoT platform with the LonWorks standard communication objects and media.

All the mechanisms, methods and solutions considered by authors in this research have been validated. Moreover, an approach to generate an expected bid for market-based demand response by using reinforcement learning mechanism is analyzed and validated by real-time operation of Energy Flexometer as well, however the study herein is limited to a small-scale application. For the sake of demonstration, two different options of Flexometer were compared (i.e., with and without nviParam). In the first option with nviParam with limited timings for appliance operation, the energy elasticity functional block learns an optimal schedule of energy and the value of price signal is lower. However, for the second one, the value of price signal increase and learn process relatively fix the time of operation as optimal schedule. This contribution is essential for effective implementation of the active DSM in EMS and provides tools for its implementation in buildings with BACS.

Future works include: (i) implementation of the developed solution in a pilot building EMS project with active DSM control and monitoring functions to verify technical concept in larger applications, (ii) development of other, additional function blocks for energy sources and storages management within DR mechanism as well as (iii) application, checking and validation of the Energy Flexometer performance in prosumers microgrids EMS.

Acknowledgments: This project has received funding from the European Community's Horizon 2020 Framework Programme under grant agreement 773717.

Author Contributions: Muhammad Babar, Phuong H. Nguyen and I. G. Kamphuis contributed to the concept and design of the research. Muhammad Babar and Phuong H. Nguyen planned and carried out the simulations. Phuong H. Phuong and I. G. Kamphuis contributed to the interpretation of the theoretical formalism and supported Muhammad Babar in analytical calculations and the numerical simulations. Muhammad Babar, Jakub Grela and Andrzej Ożadowicz conceived and planned the experiments. Zbigniew Hanzelka supervised the project and were in charge of overall direction and planning. Jakub Grela and Andrzej Ożadowicz carried out the implementation of the research, contributed to the analysis of the lab results. Muhammad Babar, Jakub Grela and Andrzej Ożadowicz wrote the manuscript. All authors provided critical feedback and helped to shape the research, analysis and the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. EIA. International Energy Outlook 2017 Overview. In *IEO2017*; EIA: Washington, DC, USA, 2017; Volume 143.
2. Papaefthymiou, G.; Dragoon, K. Towards 100% renewable energy systems: Uncapping power system flexibility. *Energy Policy* **2016**, *92*, 69–82.
3. Parvathy, S.; Patne, N.R.; Jadhav, A.M. A smart demand side management mechanism for domestic energy consumers with major HVAC load. In *Proceedings of the IEEE International Conference on Electrical Power and Energy Systems (ICEPES)*, Bhopal, India, 14–16 December 2016; pp. 504–511.

4. Ożadowicz, A. A New Concept of Active Demand Side Management for Energy Efficient Prosumer Microgrids with Smart Building Technologies. *Energies* **2017**, *10*, 1771.
5. Bahrami, S.; Sheikhi, A. From demand response in smart grid toward integrated demand response in smart energy hub. *IEEE Trans. Smart Grid* **2016**, *7*, 650–658.
6. Fan, W.; Liu, N.; Zhang, J. An event-triggered online energy management algorithm of smart home: Lyapunov optimization approach. *Energies* **2016**, *9*, 381.
7. Melton, R.B. *Gridwise Transactive Energy Framework (Draft Version)*; Technical Report; Pacific Northwest National Laboratory (PNNL): Richland, WA, USA, 2013.
8. Liu, Z.; Wu, Q.; Huang, S.; Zhao, H. Transactive energy: A review of state of the art and implementation. In Proceedings of the 2017 IEEE Manchester PowerTech, Manchester, UK, 18–22 June 2017; pp. 1–6.
9. Hu, J.; Yang, G.; Xue, Y. Economic Assessment of network-constrained transactive energy for managing flexible demand in distribution systems. *Energies* **2017**, *10*, 711.
10. Akter, M.N.; Mahmud, M.A.; Oo, A.M.T. A Hierarchical Transactive Energy Management System for Energy Sharing in Residential Microgrids. *Energies* **2017**, *10*, 2098.
11. Nunna, H.K.; Srinivasan, D. Multiagent-Based Transactive Energy Framework for Distribution Systems With Smart Microgrids. *IEEE Trans. Ind. Inform.* **2017**, *13*, 2241–2250.
12. Sijie, C.; Chen-Ching, L. From demand response to transactive energy: State of the art. *J. Mod. Power Syst. Clean Energy* **2017**, *5*, 10–19.
13. Babar, M.; Nguyen, P.; Cuk, V.; Kamphuis, I. The development of demand elasticity model for demand response in the retail market environment. In Proceedings of the 2015 IEEE Eindhoven PowerTech, Eindhoven, The Netherlands, 29 June–2 July 2015; pp. 1–6.
14. Ożadowicz, A.; Grela, J. An event-driven building energy management system enabling active demand side management. In Proceedings of the IEEE 2016 Second International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP), Krakow, Poland, 13–15 June 2016; pp. 1–8.
15. Ożadowicz, A.; Grela, J.; Babar, M. Implementation of a demand elasticity model in the building energy management system. In Proceedings of the IEEE 2016 Second International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP), Krakow, Poland, 13–15 June 2016; pp. 1–4.
16. Jachimski, M.; Mikoś, Z.; Wróbel, G.; Hayduk, G.; Kwasnowski, P. Event-based and time-triggered energy consumption data acquisition in building automation. In Proceedings of the IEEE 2015 International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP), Krakow, Poland, 17–19 June 2015; pp. 1–6.
17. Simonov, M.; Chicco, G.; Zanetto, G. Event-driven energy metering: Principles and applications. *IEEE Trans. Ind. Appl.* **2017**, *53*, 3217–3227.
18. Lobaccaro, G.; Carlucci, S.; Löfström, E. A review of systems and technologies for smart homes and smart grids. *Energies* **2016**, *9*, 348.
19. Wang, Y.; Pordanjani, I.R.; Xu, W. An event-driven demand response scheme for power system security enhancement. *IEEE Trans. Smart Grid* **2011**, *2*, 23–29.
20. Collier, S.E. The emerging Enernet: Convergence of the smart grid with the internet of things. *IEEE Ind. Appl. Mag.* **2017**, *23*, 12–16.
21. Shaikh, P.H.; Nor, N.B.M.; Nallagownden, P.; Elamvazuthi, I.; Ibrahim, T. A review on optimized control systems for building energy and comfort management of smart sustainable buildings. *Renew. Sustain. Energy Rev.* **2014**, *34*, 409–429.
22. Kok, K. *The Powermatcher: Smart Coordination for the Smart Electricity Grid*. Ph.D. Thesis, Vrije Universiteit, Amsterdam, The Netherlands, 2013.
23. Fernandes, F.; Morais, H.; Vale, Z.; Ramos, C. Dynamic load management in a smart home to participate in demand response events. *Energy Build.* **2014**, *82*, 592–606.
24. Lilis, G.; Van Cutsem, O.; Kayal, M. Building virtualization engine: A novel approach based on discrete event simulation. In Proceedings of the IEEE 2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP), Krakow, Poland, 13–15 June 2016; pp. 1–5.
25. Sučić, B.; Anđelković, A.S.; Tomšić, Ž. The concept of an integrated performance monitoring system for promotion of energy awareness in buildings. *Energy Build.* **2015**, *98*, 82–91.
26. Klaassen, E.; Kobus, C.; Frunt, J.; Slootweg, J. Responsiveness of residential electricity demand to dynamic tariffs: Experiences from a large field test in the netherlands. *Appl. Energy* **2016**, *183*, 1065–1074.

27. Mahieu, J. Active Demand Side Management. Master's Thesis, Hogeschool West-Vlaanderen, Ostend, Belgium, 2012.
28. Nguyen, P.H.; Kling, W.L.; Ribeiro, P.F. Smart power router: A flexible agent-based converter interface in active distribution networks. *IEEE Trans. Smart Grid* **2011**, *2*, 487–495.
29. Islam, S.; Woyte, A.; Belmans, R.; Heskes, P.; Rooij, P. Investigating performance, reliability and safety parameters of photovoltaic module inverter: Test results and compliances with the standards. *Renew. Energy* **2006**, *31*, 1157–1181.
30. Biegel, B.; Andersen, P.; Stoustrup, J.; Madsen, M.B.; Hansen, L.H.; Rasmussen, L.H. Aggregation and control of flexible consumers—A real life demonstration. *IFAC Proc. Vol.* **2014**, *47*, 9950–9955.
31. Abrishambaf, O.; Faria, P.; Gomes, L.; Spínola, J.; Vale, Z.; Corchado, J.M. Implementation of a real-time microgrid simulation platform based on centralized and distributed management. *Energies* **2017**, *10*, 806.
32. Camarinha-Matos, L.M. Collaborative smart grids—A survey on trends. *Renew. Sustain. Energy Rev.* **2016**, *65*, 283–294.
33. Basu, K.; Hawarah, L.; Arghira, N.; Joumaa, H.; Ploix, S. A prediction system for home appliance usage. *Energy Build.* **2013**, *67*, 668–679.
34. Camacho, R.; Carreira, P.; Lynce, I.; Resendes, S. An ontology-based approach to conflict resolution in Home and Building Automation Systems. *Expert Syst. Appl.* **2014**, *41*, 6161–6173.
35. Ivan, G.; Genesi, C.; Espeche, J.M. Deliverable-2.2: MAS2TERING platform design document. In *MAS2TERING Multi-Agent Systems Holonic Platform Generic Components*. Available online: http://www.mas2tering.eu/wp-content/uploads/2016/09/MAS2TERING_Deliverable_2.2_platform-design-document.pdf (accessed on 4 December 2017).
36. Hassan, S.; Meritxell, V. Deliverable-3.1: Multi-agent systems holonic platform generic components. In *MAS2TERING Multi-Agent Systems Holonic Platform Generic Components*. Available online: <https://euagenda.eu/upload/publications/mas2tering-multi-agent-systems-holonic-platform-generic-components.pdf> (accessed on 4 December 2017).
37. Dibowski, H.; Ploennigs, J.; Kabitzsch, K. Automated design of building automation systems. *IEEE Trans. Ind. Electron.* **2010**, *57*, 3606–3613.
38. Lehmann, M.; Mai, T.L.; Wollschlaeger, B.; Kabitzsch, K. Design approach for component-based automation systems using exact cover. In Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA), Barcelona, Spain, 16–19 September 2014; pp. 1–8.
39. Jung, M.; Reinisch, C.; Kastner, W. Integrating building automation systems and ipv6 in the internet of things. In Proceedings of the IEEE 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), Palermo, Italy, 4–6 July 2012; pp. 683–688.
40. Lilis, G.; Conus, G.; Asadi, N.; Kayal, M. Towards the next generation of intelligent building: An assessment study of current automation and future IoT based systems with a proposal for transitional design. *Sustain. Cities Soc.* **2017**, *28*, 473–481.
41. Seldon, J.R. A note on the teaching of arc elasticity. *J. Econ. Educ.* **1986**, *17*, 120–124.
42. Babar, M.; Nguyen, P.; Cuk, V.; Kamphuis, I.; Bongaerts, M.; Hanzelka, Z. The evaluation of agile demand response: An applied methodology. *IEEE Trans. Smart Grid* **2017**, doi:10.1109/TSG.2017.2703643.
43. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292.

