Implementing a Scalable Data Management System for Collected Data by Smart Meters

Farzane Farahani
Computer Engineering Department
K. N. Toosi University of Technology
Tehran, Iran
farzan.farahani@email.kntu.ac.ir

Fatemeh Rezaei
Computer Engineering Department
K. N. Toosi University of Technology
Tehran, Iran
frezaei@kntu.ac.ir

Abstract— **The Internet of things (IoT) is generating a huge amount of data and big data management is of key importance. One of the important applications of IoT is smart meter networks and one of the key issues in establishing smart meter networks is managing the large volume of data sent by the meters. In this paper, we present a data management system implemented for monitoring and managing the data collected from the smart meters and controlling them in a large-scale network. IoT infrastructure with LPWAN (Low Power Wide Area Network) class is considered in this system. Moreover, two methods are proposed to improve the performance in terms of scalability and response time. It is shown that the implemented data management system using the proposed methods achieves significant performance improvement in large scale networks.**

Keywords— internet of things, big data, data management system, smart meter network, scalability, caching.

I. INTRODUCTION

By increasing the applications of Internet of Things (IoT), a huge amount of data is generated and IoT management is of key importance. IoT management comprises device management and resource management. IoT resource management consists of process management, memory management, energy management, communication management and file management [1]. One of the key issues is data management in IoT. In data management, statistical approaches are required for analyzing and managing data. Big Data technologies are suitable for analyzing and extracting knowledge of data. Big Data in IoT-driven technologies comprises four V's; i.e., Volume (amount of data), Variety (types of data), Velocity (processing speed of data), and Veracity (truthness of data) [2].

One of the important IoT applications is in smart meter networks. A smart meter is an advanced meter which measures electricity, gas and water consumption and provides more details than a traditional one. Reading traditional meters needs spending lots of time and cost. Moreover, on a large scale, it is accompanied by errors. Smart meters networks proposed to solve these problems. It enables the process of reading customers consumption in the fastest time with the least errors and significantly reduces the costs. The smart meters network includes communication network, software and hardware for measuring, monitoring, controlling and managing the production, distribution and consumption of energy. By using smart metering on a big scale, a large amount of data will be produced and it is necessary to develop a proper data management system.

In this paper, we present a scalable system for managing and retrieving information collected from the large volumes of data in the network of smart meters. IoT infrastructure with LPWAN (Low Power Wide Area Network) [3] class is considered in this system. This wireless technology allows the connection of devices in a wide range with low power consumption and cheap connection. In this paper, comprehensive software has been implemented that has the components of a commercial product. A wide variety of features such as consumption reports, battery status, and self-check tests is provided in the developed system. It receives the meters' information such as the amount of gas and electricity consumption, voltage, current, gas temperature, and pressure at the start of every day, analyzes the collected data, and produces the desired charts and reports.

The paper is organized as follows. Section II discusses the related works. The architecture and different parts of the implemented smart meter management system are presented in Section III. In Section IV, the performance evaluation through numerical results is presented. Finally, Section V concludes the paper.

II. RELATED WORK

In IoT networks, the collected data by sensors is sent to different destinations for processing. Some pre-processing methods can be used to reduce the amount of data to be sent [1]. For instance, the authors in [4] presented a pre-processing method based on K-means algorithm for segmenting a huge number of records into clusters. The method is used for identifying and deleting abnormal consumption patterns. The purpose of this method is to determine the consumption pattern for each consumer stratum and location. The authors in [5] considered Energy Management System (EMS) which is a computer-based platform for monitoring and controlling energy usage at smart home. They suggested hybrid architecture of EMS for efficient energy management. In [6], an implementation of IoT gateway for smart metering in electrical power systems, which connects power meters with external cloud services was presented. Furthermore, they worked on smart metering applications running on IoT gateway.

Advanced Metering Infrastructure (AMI), an architecture for smart meters and data management systems, is presented in [7]. AMI enables bi-directional transmission between the

978-0-7381-2628-9/21/$31.00 ©2021 IEEE

customer and the service provider by an IP address and provides some benefits such as managing supply and demand, distributed network management and improved data quality. Moreover, [8] suggested a design of a smart metering management center for Colombia. The designed architecture has four layers: process-electric distribution system including customers, demand and generation aggregators, smart metering including functionalities inherent to the smart meter and the additional ones, metering management center composed of functions provided by Meter Data Management (MDM) and Meter Data Collector (MDC), and the group of different actors and roles of the electricity sector.

In a smart metering network, the software is a key part of it. Many researches have been done for improving the quality of the software and some methods have been suggested. After approving the meter's type, all the meters of that type should contain the same software. Moreover, checking software identity has a key role in the network security and authentication. In [9], the authors proposed methods for software identification. The authors in [10] presented a model for software quality evaluation based on Capability Maturity Model Integration (CMMI). The model consists of three layers: target layer, criterion layer and scheme layer. This method can be used for evaluating the quality of smart electricity meters with uniform specifications from different manufacturers. In addition to evaluating the quality of the software, it is necessary to evaluate the reliability of the software. In [11], the authors presented the analytic hierarchy process model for reliability evaluation, and proposed two schemes, one for performance comparison among different functions in the same meter, and another for comparing different meters of different manufacturers. They have combined the schemes in order to analyze the performance of different types of smart meter software and different function modules in the same smart meter software.

III. The Proposed Smart Meter Management System

In this section, we present the architecture and different parts of the proposed system for monitoring, managing and controlling the smart meters. The implemented computer-based platform has various components and features which are explained in the following.

A. The System Architecture and Components

1. Smart meter installation

When new customers join the network, it is required to install new smart meters for them. Also, it is necessary to change the old meters with new ones. For installing a new meter, a request has been sent to the server. By receiving the meter information such as serial number and user id, the meter will be registered and gets ready for remote management.

2. Provide billing data

Based on customer consumption, daily, weekly and monthly reports are generated and saved to the database for future analysis and decision makings.

3. Checking meters

When some alarms about manipulations and other kinds of failures are sent by the meter, it is necessary to visit the meter at the location. After visiting, the results will be recorded in the database.

4. Destruction and renovation

When the destruction and renovation message is received from the customer, the consumption reports will be generated and the meter data will be removed from the database and then the meter will be disconnected.

5. Checking manipulations

When the meter detects manipulations such as meter movement, cables disconnection, etc., the manipulation cases are provided to the management system and related alarms are produced.

6. Meter displacement

The process gets started when a customer sends a request to change the meter location. After confirmation, the consumption reports get generated and then the meter gets disconnected and can be moved.

7. Reading on request

Meter reading is required in different situations such as checking meter operation, complaints handling, etc. For meter reading, a request is sent to the server and the related data will be returned.

8. Messages and alarms management

This process includes the management of alarms and events related to the meter operation and efficiency. The warnings include low battery, overconsumption, gas valve status changes, etc.

9. Detection of non-connection to the meter

The management system must be aware of the communication status with the meters in order to be able to manage the meters properly.

10. Setting meters time

The consumption measurement accuracy depends on the accuracy of the meter clock. For this reason, the meter clock must be synchronized with the national standard clock. After sending time from the management system to the meters, synchronization is performed.

11. Self-check tests

Self-check tests such as battery status tests, meter connection tests, etc. are used to ensure the correct operation of the meters. In addition, to run tests on demand, they are run in the specified intervals as scheduled.

12. Request meter information

In some cases, the meter information such as serial number, meter body number, etc. is required. To access the information, a request is sent to the server and the information will be returned if the meter exists.

B. Software Implementation

Since Python is one of the most powerful and strong programming languages, we have used one of its high-powered frameworks called Django which is free and open source. It provides rapid development and clean, pragmatic design. Moreover, Django takes security seriously and provides the ability of scalability as flexible as possible [12].

One of the industry-standard web development frameworks to create scalable and extensible projects is Model-View-Controller (MVC). Django at its core is based on MVC architecture. It actually implements Model-Template-View (MTV) architecture which is a bit different from MVC architecture. Both are the same in two

components called model and view. The model contains logical file structure of the project and also acts as a data handler between view and database. The view is formatting the data via the model and also it communicates with the database and the data that is transferred to the template for viewing. The template is the only component that distinguishes the MTV pattern from the MVC pattern. It is like a presentation layer that handles the user interface part completely. Keeping whatever the browser renders is the main goal of the template. Generally, in MVC pattern we need to write all the specific control code however in MTV pattern, it is done by the framework, so the MTV pattern makes developing faster. Also unlike in MVC, view is not coupled with a model and this makes MTV loosely coupled and easy to modify [13].

C. *The Implemented Smart Meter Management System*

In this section, we present the implemented system. Fig. 1 shows the home page which includes the regions equipped with smart meters. By clicking each region card, the list of the meters belonged to the region is accessible. On a specific meter page, it is possible to check the detailed information of the meter, the consumption reports, gas pressure and temperature, voltage and current level, battery status, etc., as illustrated in Fig. 2. Moreover, it is possible to manage the messages of the meters as shown in Fig. 3. The alarms related to manipulations, such as changing meter location illegally and disconnecting cables, which have been sent by the meter can be accessed on this page. Also, the messages that are sent by the management system to the user are illustrated in Fig. 4. In addition, it is possible to install a new meter by entering the meter detailed information in the specified form. The possibility of deleting a meter entity or part of its data is also provided in the software that is shown in Fig. 5. By using the meter number and the user ID, the user can access the meter's information and all related data for making some changes if necessary.

D. *Scalability and Performance Improvement*

The software plays a key role in smart metering network and the quality of the software is of particular importance. The performance of the presented software mainly depends on the database capabilities. The system must be able to handle the workload fluctuations properly. If the databases are not scalable, there would be lots of performance problems in processing user queries. Database calls are costly and the number of database trips for answering user queries plays a key role on the software performance. Therefore, the following methods are used so that the database can handle large volumes of data.

One of the effective methods to improve the database performance is caching. A suitable caching strategy reduces the overhead on the database and improves the software performance and reliability. As database calls are expensive, the frequently used data could be kept in the cache in order to deny so many database trips. Indexing is another method to improve the performance of the database. The purpose of indexing is to access needed data with less search. It is the same as making a table of content for a book to find a particular part without checking all of it. Indexes make it possible to retrieve data more quickly without reading the entire table. Indexes are not free and they use some space, so it is important to know where to use them [14].

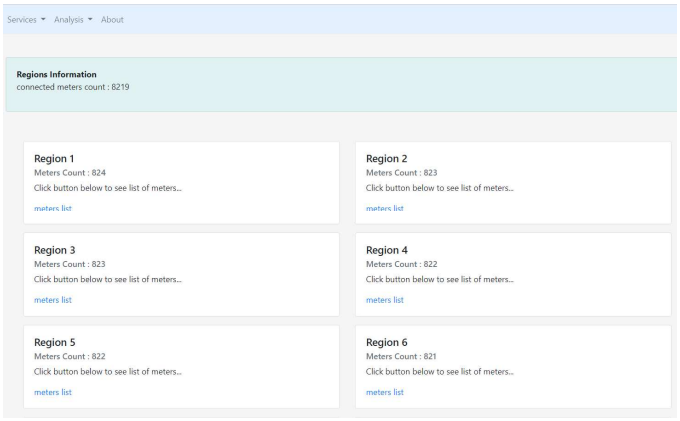

Fig. 1. The software homepage

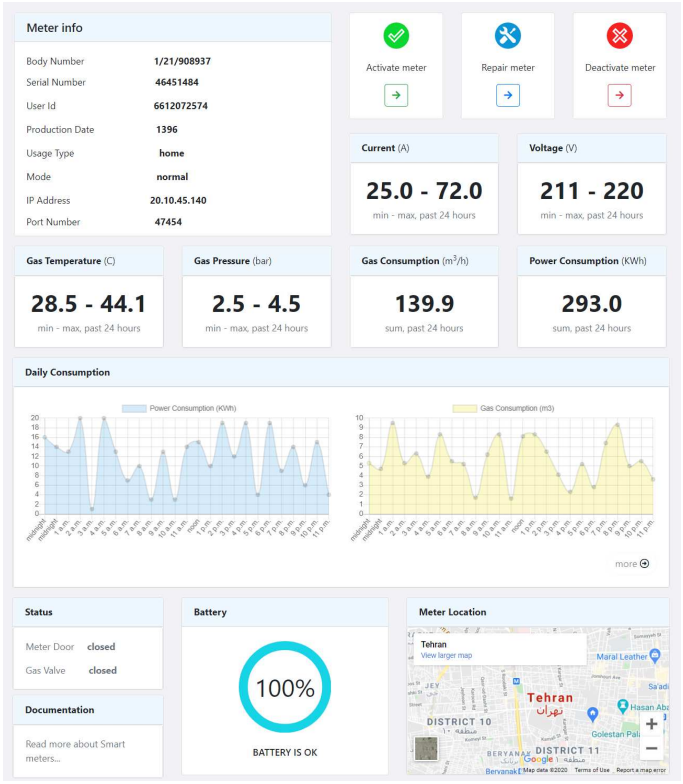

Fig. 2. The data of a specific meter

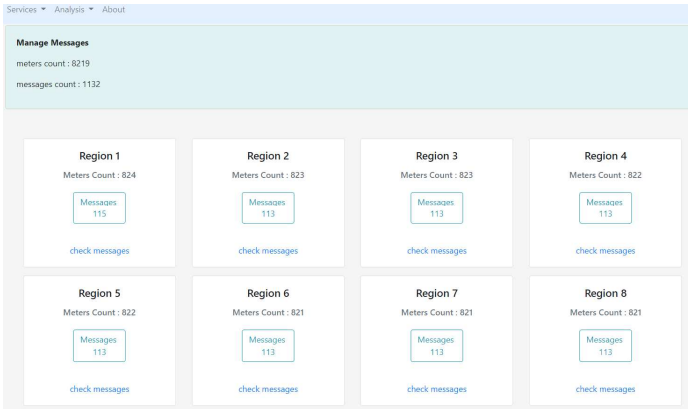

Fig. 3. The messages and alarms page

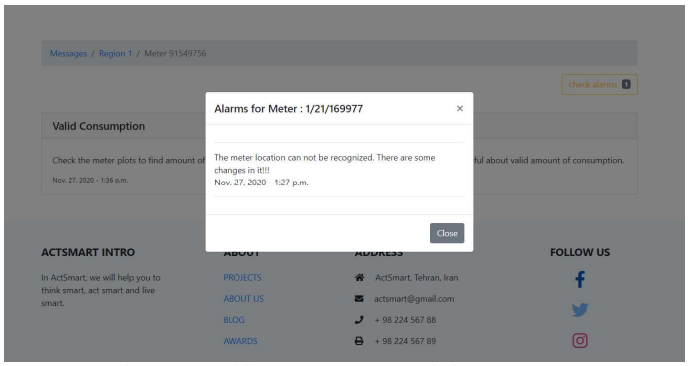

Fig. 4. A specific meter messages and alarms page

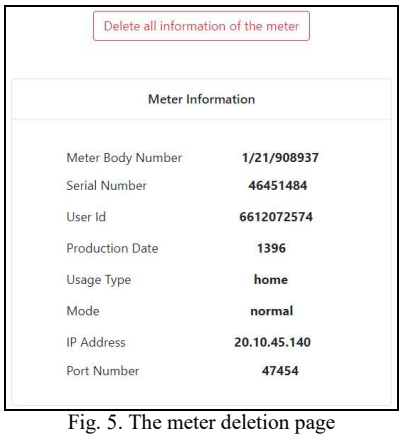

Fig. 5. The meter deletion page

Moreover, proper data preprocessing improves software performance. For example, checking duplications in the data sent by the meters can reduce the data volumes that must be stored in the database. For implementing data deduplication, we use comparing the chunks of data to detect duplicates, and hence some unnecessary records are removed in this way. Moreover, to validate the data stored in the database, we delete unauthorized data which is the data belongs to the meters without a verified number. To achieve this goal, we check the structure of the meter numbers and remove the data related to the meters with unacceptable number.

E. Comparison of the System with the Previous Ones

The systems that are designed to manage smart meters must meet three major requirements: improving and optimizing network performance, improving and optimizing utility management, and enabling customer engagement [15]. According to the description of our system, it includes all three mentioned features.

The authors in [16] presented an energy management system software to achieve the aim of energy saving. In this system the same as ours, through the energy consumption report of energy management system, the users can grasp all kinds of energy consumption. Through energy consumption indicators such as unit area, energy consumption, and other indicators, the abnormal values of energy consumption can be found out. By using this system, they achieved reducing energy consumption but as they stated, there are still some deficiencies in the development of their system. For example, the memory configuration of the database should be optimized and its use performance should be improved as we have paid attention to these tasks in our system.

IV. Performance Evaluation and Results

In this section, the performance of the implemented software is evaluated through numerical results. For the data collected from meters, we have considered JSON format which is a self-describing and lightweight format that suits for transferring and storing data. The information such as the amount of gas and electricity consumption, current and voltage level, gas pressure and temperature, the gas valve status, the meter's door status and the battery status. The received data messages are decoded into variables that are defined in the program to be used in the related parts for managing the meters. We evaluate the performance of the implemented software and the basic methods affecting it. Figs. 6-8 illustrate the response time of the select, insert and update queries for different number of smart meters.

As can be seen, the response time highly depends on the number of the smart meters and the amount of data stored in the database and database capabilities. When we are planning to use a method to improve the performance of the database, we should consider the main operations performed on the database and choose the appropriate method. For instance, although indexing is effective in searching operations, it may experience a slowdown in the insertion operation. As can be seen in these figures, most of the operations performed on the database in the developed software are of the search type. Therefore, in order to improve the average response time and to find the needed data more quickly, we have created some indexes for the consumption table. Since many records are belonging to a meter, an index is created for each meter column. This method accelerates functions such as billing. In addition, we create an index for the date column, because it is helpful for deleting and filtering the desired data. For each index, a B-tree is created which provides an efficient way to read and insert data. Using the indexes makes the database perform a binary search instead of reading all records to find the desired one and improve the performance. The issue that should be considered here is the number of the created indexes. As discussed earlier, the indexes require the storage space and need to be updated when new records are added. Therefore, indexing should be used wisely in the necessary cases in order to improve the performance.

Moreover, for some cases that indexing cannot help effectively, we perform database caching for the indexed databases. In the developed LRU caching method [17], [18], the cached data is stored in a table created for caching in the database, in order to use the cached data in necessary cases and reduce the database calls. It happens a lot that the user calls the same query to fetch data. Since the database is updated every morning due to the information sent by the meters and not many changes are made during the day, we cache the Query Set which is returned by a particular query. The arguments that are used to control the caching behavior are timeout, the maximum number of entries allowed in the cache, and the number of the culled entries when the table reaches the max number of entries. The timeout value is set to 24 hours, since the database updates once every day. The other point to be noted is the amount of data in the cache table, which without controlling will cause cache malfunction. The maximum number of the entries allowed in the cache is set to 100,000 to avoid having a big table. This number is considered due to the extent of the system and it can be reduced or decreased based on the system requirements after deploying the system on a large scale.

Moreover, the number of the entries that are culled when the table reaches the max number of entries is set to half of the table records.

In Fig. 9, we have compared the response time of the select query before and after implementing the caching method on the indexed database. As it is shown in this figure, by increasing the number of the smart meters, the number of the database calls has been decreased and the caching method on the indexed database has made searching faster. Therefore, the implemented data management system performing the proposed method achieves significant performance improvement in large scale networks.

V. Conclusion

In this paper, we have presented the software developed to manage smart meter networks remotely. Moreover, we have implemented two methods to improve scalability and database performance. It has been shown that implementing the proposed methods in large scale networks decreases the response time and improves the performance.

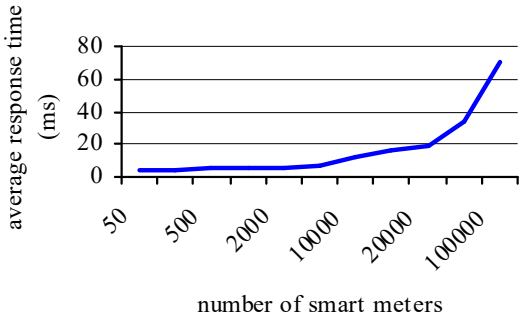

Fig. 6. Average response time of the select query

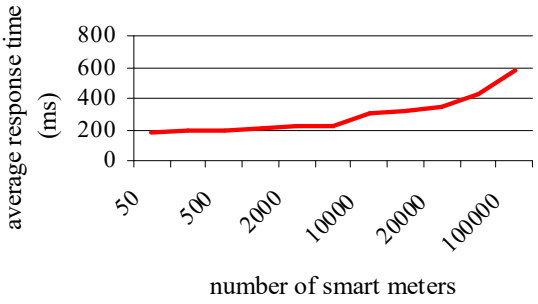

Fig. 7. Average response time of the insert query

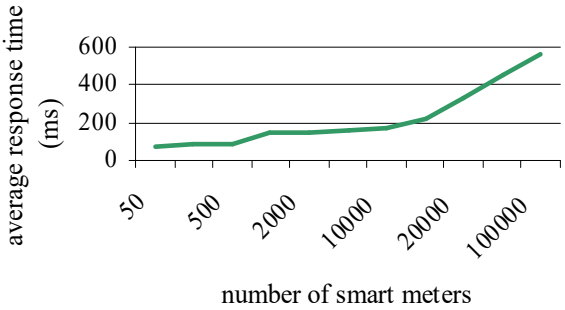

Fig. 8. Average response time of the update query

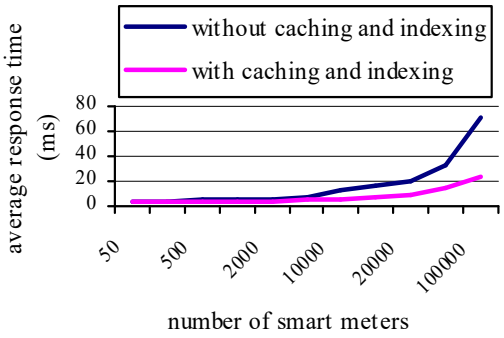

Fig. 9. Average response time of the select query

References

[1] S. Sinche, and F. Boavida, "A Survey of IoT Management Protocols and Frameworks," *IEEE Commun. Surveys & Tutorials*, vol. 22, no. 2, pp. 1168-1190, 2020.

[2] S. Khare and M. Totaro, "Big Data in IoT," *Int. Conf. Computing, Commun. and Net. Technol. (ICCCNT),* Kanpur, India, 2019, pp. 1-7.

[3] K. Mekki, E. Bajic, F. Chaxel and F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT Express,* vol. 5, pp. 1-7, 2019.

[4] S. Celis, L. F. Giraldo, P. D. Oliveira-De Jesus and C. A. Rincon, "A Clustering Approach for Domestic Smart Metering Data Preprocessing," *IEEE ANDESCON*, Santiago de Cali, 2018, pp. 1-3.

[5] T. Ku, W. Park and H. Choi, "Energy information collecting agent for IoT big data system," *Int. Conf. Inf. and Commun. Technol. Converg. (ICTC),* Jeju, 2017, pp. 1134-1136.

[6] M. P. Shopov, "IoT gateway for smart metering in electrical power systems - software architecture," *Int. Conf. Inf. and Commun. Technol., Elec. and Microelec. (MIPRO),* Opatija, 2017, pp. 974-978.

[7] S. Nimbargi, S. Mhaisne, S. Nangare and M. Sinha, "Review on AMI technology for Smart Meter," *IEEE Int. Conf. Advances in Elec., Commun. and Comp. Technol. (ICAECCT)*, Pune, 2016, pp. 21-27.

[8] W. M. Salamanca and J. R. Garcia, "Metering Management Center for Colombia and short-term strategies to implement programs related to smart metering," *FISE-IEEE/CIGRE Conf. Living the Energy Transition (FISE/CIGRE),* Medellin, Colombia, 2019, pp. 1-7.

[9] H. Huang, L. Wang, Z. Jia and Y. Pan, "Software Identification for Smart Meters," *Conf. Precision Electromag. Measur. (CPEM 2018)*, Paris, 2018, pp. 1-2.

[10] L. Wei, W. Yong, H. Huijuan and Z. Xiaoqi, "Software Quality Evaluation Model of Smart Electricity Meters Based on CMMI," *5th Int. Conf. Comp. and Commun. Syst. (ICCCS),* Shanghai, China, 2020, pp. 157-161.

[11] Z. Feng, L. Xiaobing and X. Bin, "Reliability evaluation technology research for smart meters software," *IEEE Conf. Inf. Technol., Net., Elec. and Autom. Control,* Chongqing, 2016, pp. 888-892.

[12] Django software foundation and individual contributors 2020, <https://www.djangoproject.com >

[13] Data Flair 2020, < https://data-flair.training >

[14] R. Ramakrishnan and J. Gehrke, *Database management systems,* New York, USA: McGraw-Hill, third ed., 2003.

[15] D. Avancini, J. Rodrigues, S. Martins, R. Rabelo, J. Al-Muhtadi and P. Solic, "Energy meters evolution in smart grids: A review," *Elsevier J. Cleaner Production, vol. 217*, pp. 702-715, 2019.

[16] H. Zhang and Z. Zhou, "Design and implementation of energy management system software in green building," *Acta Technica,* no. 1A, pp. 495-506, 2017.

[17] F. Rezaei and B. H. Khalaj, "Stability, Rate and Delay Analysis of Single Bottleneck Caching Networks," *IEEE Trans. Commun.,* vol. 64, no.1, pp. 300-313, Jan. 2016.

[18] F. Rezaei, A. Momeni, B. H. Khalaj, "Delay analysis of network coding in multicast networks with Markovian arrival processes: A practical framework in cache-enabled networks", *IEEE Trans. Vehicular Technol.,* vol. 67, no. 8, pp. 7577-7584, 2018.