GU J Sci 31(2): 472-487 (2018)

Gazi University

Journal of Science

http://dergipark.gov.tr/gujs

A Study of Development of Standardised M2M Service Platform Using Restful Architecture

Huseyin POLAT[1], Saadin OYUCU[1,*], Necaattin BARISCI[1]

[1]*Department of Computer Engineering, Faculty of Technology, University of Gazi, 06500, Ankara, Turkey*

Article Info

Received: 05/10/2017
Accepted: 26/02/2018

Keywords

M2M platform
Web services
NoSQL
SOA
RestFul

Abstract

Machine to machine (M2M) is a smart ecosystem that enables devices to communicate by exchanging data through specific protocols autonomously with limited human interaction (or no human interaction) between each other. Use cases, business models, varieties of devices and protocols of M2M systems getting proliferating. Being able to accommodate this large spectrum of possibilities requires a standardized M2M platform design based on abstraction, modularity, and scalability. In this study, a web-based scalable M2M platform was developed which is compatible with European Telecommunications Standards Institute (ETSI) and oneM2M standards. The web-based M2M service platform has with a sensitive design adopting Web 3.0 standards and Service Oriented Architecture (SOA) approaches. The access to M2M services and data was provided using Representational State Transfer, RestFul Web Services (REST) through a service layer. The data acquired from M2M devices were stored in a non-relational database. The security of M2M platform was achieved through token based ID check. Fast, scalable, cost-efficient, secure, standardized, general purpose an M2M service platform was created.

1. INTRODUCTION

Machine-to-Machine (M2M) communication refers to the technology that enables wired or wireless communication between different devices [1]. International Telecommunication Union (ITU) defines M2M as at least the communication of two machines with or without minimum human interference [1]. On the other hand, M2M platforms are web-based platforms that are developed through M2M technology in order to ease the management of applications which serve to different industries [2]. All management requirements such as remote monitoring systems, measurement analysis, and reporting are available to users through M2M platforms. By means of these platforms, the operational power of M2M devices has been increasing.

In order to develop M2M applications, the M2M platform is needed. M2M platforms should be designed to serve different users and devices simultaneously. M2M platforms use various communication technologies including wireless networks technologies such as cellular networks, WPAN, RFID, Wi-Fi, WiMax and satellite networks [3]. The advancement of IP-based network technologies led to the improvement and more widespread use of M2M systems. This paves the way for the end users to develop M2M applications. M2M devices are not enough on their own for M2M applications, there is also a need for the transfer of data to M2M platform and tracking the data on this platform. In order to develop an M2M application, detailed web technologies knowledge is required. Through the M2M platform developed in this study, end users are provided the opportunity of easily developing M2M applications. The infrastructure for the essential information technologies for M2M applications is supplied with the M2M platform.

*Corresponding author, e-mail: saadinoyucu@gazi.edu.tr

When the literature is reviewed, it is found out that the previously developed M2M platforms used different communication techniques, service methods, and database systems. In the majority of the studies in the literature, M2M platforms are developed on smart home systems. Zhang et al. in their platform design study on M2M [4], tried to develop a system using Java Enterprise Edition (J2EE) and SOA approach. In the system, which is developed as service oriented, Extensible Markup Language (XML) was used as messaging language. Castro et al. conducted an M2M platform analysis study. In their study, they investigated platforms such as RunMyProcess and Axede. According to the study, it was revealed that the majority of the platforms was developed using SOAP and XML [5]. Meddeb et al. conducted their study OMA DM, a distant management protocol, through analyzing ETS and oneM2M architectures. They used autonomous devices under the scope of their study. Device management was recommended to be autonomous. They made use of SOAP and XML in the development of the platform [6]. Youm and Kim in their study of developing a prototype of the smart home project [7] through using M2M platform structure which provides horizontal service integration, used open source Connected Objects Operating System (COOS). Song et al. in their smart home and campus study developed a system which they named as SENSEI. In their study, they made a modeling of M2M system using Rest web services and XML [8]. Takatsuka et al. in their study developed a platform named as Rucas [9]. In the development of Rucas platform, instead of MongoDB as database and REST as web services, they used XML and SOAP technologies. Alaya et al. developed a platform which they named as OM2M. They tried to add self-configuration skill to the M2M platform which was compatible with ETSI [10].

In some other studies, the use of web service standards in devices was recommended in order to enable interoperability in the whole system. These standards generally focused on SOAP and WSDL concepts through applying Service Oriented Architecture (SOA) [11-12]. Some of the studies, in which web services were used, are about Internet of Things (IoT). In IoT studies, REST was preferred instead of SOAP. However, XML was preferred as messaging method [13-14].

In this study, a user-friendly, scalable, secure, high performing and having a layered architecture M2M platform was developed. Different from the literature, during the design of the platform Rest web service approach was preferred rather than SOAP, and JSON used as a messaging method instead of XML. Additionally, a document based, the non-relational database was preferred. Through this design, the data received from M2M devices in JSON format are saved in the database in JSON format without making any changes. In this way, the unnecessary conversions between XML-JSON or XML-database were eliminated. This provided advantages in terms of ease of use, cost efficiency, time, and speed.

2. MATERIALS AND METHOD

2.1. M2M Standards

The standardization in method and architecture has turned out to be of high importance due to the high number, complexity, and security of applications developed on M2M. ETSI and OneM2M are the pioneers of the development of these standards. ETSI which developed standards on M2M until the year 2012, has many studies were done and produced standards on smart homes and home energy management. On the other hand, in the year 2015, OneM2M organization published the first worldwide M2M standards 2015 [15].

2.2. M2M Architecture

According to OneM2M standards, M2M architecture is composed of three main layers as basic structure [16]. OneM2M's M2M architectural structure, which is published in the document explaining M2M architectures and more functional compared to other architectures, can be seen in Figure 1.

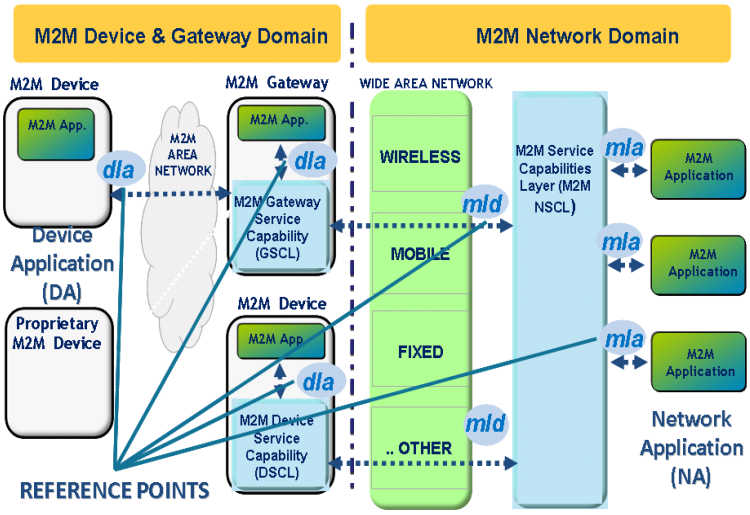

Figure 1. *Functional M2M architecture [16]*

The architecture divided into two which are M2M device & gateway field and M2M network field has also different subfields under each field. Especially different from the basic architecture, the scalability is increased through implementing M2M service abilities layer. Additionally, the development of different applications is made easier. The requests from M2M applications and M2M devices are interpreted in the M2M service abilities layer and the requested transaction is done. The system can easily be scaled through either extending functions or adding new functions to the service layer. Therefore the M2M platform developed in this study was based on the architecture in Figure 1.

2.3. The Use of XML and JSON

The data should be presented to each system and user in the form of standard data structures. Therefore XML data structure is generally used. However, the increase in structured, semi-structured and unstructured data and the emergence of big data have replaced XML with JSON. JSON, whose popularity is increasing day by day, perfectly works especially in the web services designed specifically for it.

Table 1. *XML and JSON data format*

Sample XML Structure
<? xml version="1.0" encoding="UTF-8"?> < device> < deviceID>2e6e0c8b-12c7-4353-be5c-7a2486c52db8</deviceID> <device_name>Okul DHT11</device_name> <userID>55227aed88e5f382b2006c59</userID> <token>3e500d06-6c53-444a-8e81-18abc99fb34f</token> < humidity>35.00</ humidity> <temperature>21.00</temperature> </ device>
Sample JSON Structure
{ "userID" : "55227aed88e5f382b2006c59", "deviceID" : "2e6e0c8b-12c7-4353-be5c-7a2486c52db8", "token" : "3e500d06-6c53-444a-8e81-18abc99fb34f", "cevice_name" : "Okul DHT11", "temperature" : "21.00", "humidity" : "35.00" }

In M2M platform different users may make more than one requests or transactions at the same time. In such situations, M2M platform is required to give fast responses to the user. For this reason, the data needs to be presented faster and in a smaller size. During the process of data identification in XML structure, the use of title tag is obligatory (Table 1). However, in JSON structure this process can easily

be done through punctuation marks (Table 1). There is a need to use XML converters while composing XML tags and saving the data received through XML. When JSON data structure is used there is no need of such a conversion. However, in order to achieve this, a non-relational database which supports JSON data structure, and appropriate web services need to be used. Through this usage, complexity of M2M platform is lessened while its speed has increased. JSON structure not only eliminates the complexity but also reduces the size of the data. Thus, the data that will be transferred through the network is carried in JSON data structure in a smaller size than XML.

Table 2. *The size comparison of XML and JSON data structures*

Piece	XML File Size	JSON File Size
1	314 Byte	220 Byte
10	2.780 Byte	2.173 Byte
50	13.748 Byte	10.856 Byte
100	28.458 Byte	21.756 Byte

In this study both because of its plainness and simple, easy-to-understand structure, JSON data structure is used. When table 2. is examined, it can clearly be seen that the JSON data structure is in smaller size during the transfer over the network compared to XML. For this reason, in order to use the network traffic more efficient, JSON data structure is preferred for the M2M platform developed.

2.4. Choosing The Database

The data obtained from M2M devices and stored in M2M platforms can reach to quite large sizes. Certain problems emerge in the process of writing, reading and analyzing the large amounts of data in the database. When the previous M2M platform studies are investigated, it is found out that relational database model is used to store the data. Structured Query Language (SQL) which is known as relational database query language, is also known as a database [17]. The reason behind the high importance of SQL Databases and its use in large projects are their quite efficient background support. However, nowadays, the spread of Cloud Computing and distributed web applications have led to an increase in the need for databases whose usability and scalability are high. Thus, the concept of Not Only SQL (NoSQL) has emerged. Nevertheless, in some of the NoSQL databases, there exits such limitations as reporting and support of SQL standards. On the other hand, the advantages of NoSQL databases are high reading and writing speed, support of bulk data transactions, ease of expanding and being low-cost [18].

The data are received from one or more M2M devices to the M2M platform. During the writing process of these data to the database, sometimes there may be a high workload. During the reading of the data from the database, more than one user may send different requests to the database. If the duration of database's response extends, database lockdown may be faced. In order to prevent this, a high performing document-based NoSQL database should be preferred. MongoDB is a self-proved system. MongoDB is a document based NoSQL database. MongoDB presents many features as document data model, rich query support, horizontal scalability, high usability, flexibility, and dynamic schema. In the data storage process, MongoDB allows the use of JSON data structure. The systems in which NoSQL database and RestFul web services are used, enable development of more dynamic and applications independent from infrastructure. In this study, MongoDB is preferred, for it supports JSON data structure.

2.5. Web Services

Hypertext Markup Language (HTML) has a static structure. However, nowadays dynamic web pages and documents are preferred. The web pages developed through Microsoft's Active Server Pages (ASP), Java's Java Server Pages (JSP) and Hypertext Preprocessor (PHP) codes have been replaced with static web pages developed with HTML [19].

Service Oriented Architecture (SOA) is basically a software architecture that is developed for using web services. SOA encourages autonomy of non-isolated individual logic units. While the logic units are preserved with enough amount of partnership and standardization, they must be compatible with a series

of principles that allow independent development. In SOA these logic units are known as services. The analysis of automatization which is composed of services that can be seen in Figure 2. Each service may contain one stage or a task which is done through a sub process including a series of stages. Moreover, a service may also involve the whole system logic [20].

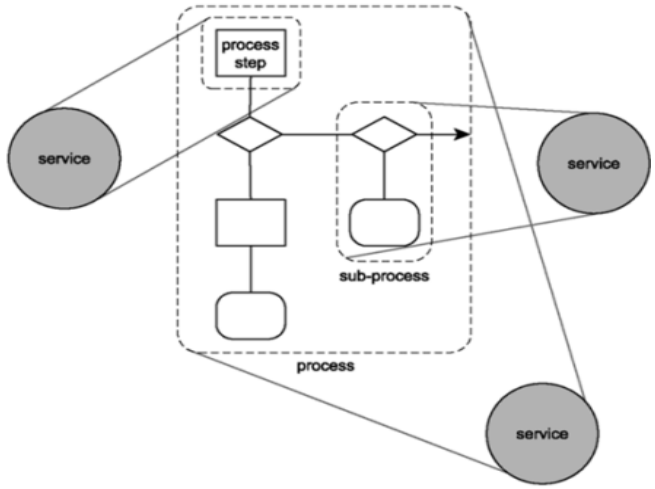

Figure 2. *Sample SOA automatization analysis [20]*

Services are known basically as software parts that can send requests to different applications and devices. Through the distributed architecture more than one application can be brought together and presented to the user. Web service development architectures present a model that represents automatization logic by separating small and different logic units. These units as a whole contain a large part of work automatization logic [20]. In order to enable the use of web service technologies, three agents need to contact with each other. These agents are service provider, service client, and service registry unit. Many standards have been developed on this structure; but basically, Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Description, Discovery and Integration (UDDI) standards are preferred. Web services have become stronger with the use of concepts that can work together such as HTTP and XML.

RestFul approach is another service architecture approach which has recently become popular and gradually increased its fields of use. This concept which emerged after the publication of Roy Fielding's dissertation [21] in 2000 is an architecture that is used with a series of criteria. Fielding determined certain criteria for REST [21]. These criteria are composed of 6 units which are Client-Server, Stateless, Cache, Uniform Interface, Layered System, and Code-On-Demand. REST architecture basically resembles SOAP; however, it is an easier –to understand structure than SOAP since it is based on the simple request-response principle between client and server. This simple request-response structure provides convenience for Application Programming Interfaces (API). RestFul structure does not depend on any technology. It provides XML and JSON messaging structure. In RestFul architecture, four basic operations of HTTP are used. These are GET/POST/PUT and DELETE. GET is used in the process of receiving the data without changing its structure in the resource while PUT and POST are used in the process of receiving data by making changes in its structure in the resource [22].

In this study, SOA is used as architecture and RestFul web services that are chosen as a web service approach. The reason is that JSON structure which is supported by RestFul web services has many advantages over XML. Because of such features as a layered architecture, caching, server-client structure, and not keeping the session information in the status info, RestFul structure is preferred in this study. These facilities provide ease of use and high performance in M2M applications. Additionally, the preference of JSON messaging structure, the use of JSON supporting web services, and the use of NoSQL database system in which the records can be kept as JSON provides an advantage in favor of the M2M platform. In the M2M platform developed, only JSON data structure is used and since there is no conversion in the structure of data, a considerably high speed has been achieved.

2.6. M2M Device and Sensors

In this study, Arduino card is used to develop a prototype M2M device. Arduino card is a good choice in terms of the ease of use and cost efficiency. Arduino card with a processor which has a microcontroller that can make 8-bit transactions can be found in different models. Arduino has 8-16 MHz speed and 2-8 KB RAM capacity and it can easily be programmed through its coding application that can work in different operating systems. Through Arduino, various sensors as moisture and heat sensors can be used. In this study, LM35 heat sensor and DHT11 heat-moisture sensor are used. In order to communicate with the M2M platform developed on IP networks, Arduino Ethernet shield is used.

2.7. Interface Standards and Technologies

The issues as how the data will be presented to the user in M2M platforms and how the identity information will be received by and from the users are highly important. Moreover, the M2M platform should be user-friendly. Therefore in the development of M2M platform, new application development techniques, methods and design standards are used. Especially HTML version 5's Cascading Style Sheets (CSS), version 3, and JQUERY library are used in the M2M platform. Additionally, platform interfaces are presented to the users with a sensitive design. Therefore an interface was developed that automatically adapts to mobile systems, smartphones, tablet PCs, laptops, and desktops.

3. DEVELOPMENT OF M2M PLATFORM

3.1. M2M Platform Architecture

In this study, the architectural structure of M2M platform was formed based on the international standards. Traditional M2M architectural structure and the M2M architectural structure that are used in this study can be seen in Figure 3.

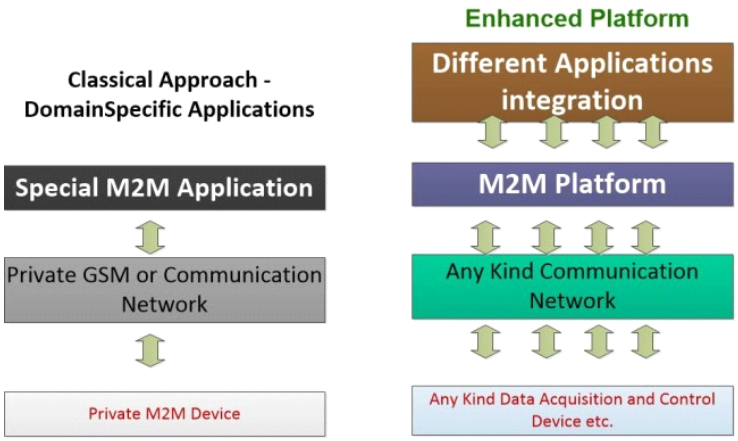

Figure 3. *Traditional M2M architectural structure and M2M architectural structure used in this study*

In traditional M2M architecture, field specific M2M device, GSM or communication network is available. In this structure, a specific M2M application is developed and presented to the user. This approach depends on the person and field of use. Moreover, for each agent change, the whole architecture has to be designed from scratch. In the architecture used in this study any data collection, control unit, or device can be used. As for communication, a network technology that can perform an IP-based communication can be used. M2M platform is designed in a way that can serve simultaneously to different devices, different users from different devices (tablet PC, PC, smartphone etc.). In the M2M architectural structure used in this study, there is a service abilities layer which is not available in the traditional architectural structure. Service abilities layer is the one that meets the requests from different users and M2M devices. Service abilities layer interacts with the user with the help of an interface and in the background, it is in constant communication with the database system. In this way, the feature

asserted by Roy Fielding that user should not reach the information about the source of data is enabled [21].

3.2. M2M Platform Database Transactions

In this study, MongoDB NoSQL database which operates in document based principle is used. It supports JSON structure. The M2M platform developed works on a Windows based operating system. Therefore the configurations and studies are conducted on Windows-compatible MongoDB. Since MongoDB is a document based system, it basically runs and starts through codes rather than interfaces. In order to run MongoDB, a user who has administrative authorization is required. In the system prompt screen, the database should be made ready with the help of codes. Therefore the code below should be applied.

c:\development\db\mongodb\bin\mongod —dbpath c:\development\db\data --port 27017 –rest

When the above command structure is examined first, the location of documents that are required to run MongoDB structure is indicated. Thus the system is made ready through running "mongod". The location where the data documents will be saved should also be indicated in the command. Therefore in the above code fragment the location that the documents will be saved is indicated through "--dbpath" command as in the directory near the command. From its start, MongoDB reserves 27017 Transmission Control Protocol (TCP) port for itself. In order for streaming at a stable port with the "--port" command, the number of the port that is required to stream in is specified. MongoDB provides different techniques when compared to web service architectures. By adding "--rest" command on the code fragment above, the database system is requested to be ready for the requests received over simple HTTP.

MongoDB stores each data by adding a unique Identity (ID). The user cannot directly reach these stored data as JSON document but rather has to use assisting tools. While some of these are standalone interface tools as RoboMongo, others are software as Eclipse MonjaDB etc. that are integrated into application development software. The mentioned software as in SQL database lists the data as lines and columns and presents users. While the data is kept n tables in SQL databases, in NoSQL the concept of the table is replaced with the collection. Under normal circumstances, raw data is kept as JSON in MongoDB database. After these processes, the database is made ready. The next step is to build web service layer.

3.3. Preparing M2M Platform Service Layer

In the M2M platform developed RestFul web services are decided to be used. There exist many structures for building RestFul services. Spring ready structure is the leading of these structures. Spring provides a comprehensive configuration model for Java-based programming and applications [22]. Among the Spring versions, 3.0 supports Rest architectural structure [23]. In the M2M platform developed, Spring version 3.2.3 is used. In order to increase the transferability of the developed software, Spring has two different methods. The first of these is "Maven" projects, and the second is "Gradle" projects [23]. Since the development and use of "Maven" projects are easy, this study is started as a "Maven" project. During the development of the project sometimes the need to integrate different architectural structures may be needed. This need can easily be fulfilled by adding features from "prom.xml" folder which is available in Maven projects.

In order to develop the M2M platform, the software context should be successfully prepared. The software needs of M2M platform operating on Windows operating system are; Eclipse Kepler, Java Standard Edition Version 8 and Apache Tomcat 7.0 server. Eclipse Kepler version perfectly suits for developing web software and web services. Due to having different toolbars and supporting Maven projects Eclipse Kepler version is used in this study. The reason why Java version 8 is chosen is that it works perfectly with Spring version 3.0 and 4.0 as well as it supports NoSQL connection structures. In order not to experience complexities during the development of M2M platform, all of the configurations are brought together under the same directory with the database.

3.3.1. Generating framework for web services

When the maven Project first started "pom.xml" document is encountered. Developer keeps the project configuration and necessary dependencies in this document. On the top of the XML document the project information is defined. The versions of extensions used are determined after determining Maven version. With the "dependency" tag, the documents with .jar extension are downloaded to the local computer, so that the need to connect the remote computer every time when the system starts. "groupId" shows the name of the software package while "artifactId" showing the name of the project. As it is obvious the information necessary for the system is defined in "pom.xml" document. The new extensions, features, and libraries can be integrated into the system from this document. The tests, database features and all other necessary abilities for M2M platform can easily be adapted to the structure.

3.3.2. Database connections and setting the basic model

While the web service framework was created, Spring web MVC feature was integrated into the system. Model-View-Controller (MVC) -to define shortly- is a structure that enables to make web based projects based on Spring. The required additions according to the database used in the platform are available on the framework. The only work to be done is making the connection and check. For this, the necessary "MongoDbFactory" interface connectors should be used.

Each process in the design of software is done under a package and in classifications. Therefore this project gained transparency. After the connection to the database is enabled, the next process is setting models which will be used for all database transactions. For instance, when a new record is added to the database, a date of record is also added. Such repetitive transactions are defined in the basic model. To set an example, the only creation date is added to the basic model. In this way, flexibility and plainness are gained.

3.3.3. Defining services

In RestFul architecture request or responses realizes in a certain order. The Spring MVC RestFul architecture which is used for service development is shown in Figure 4.

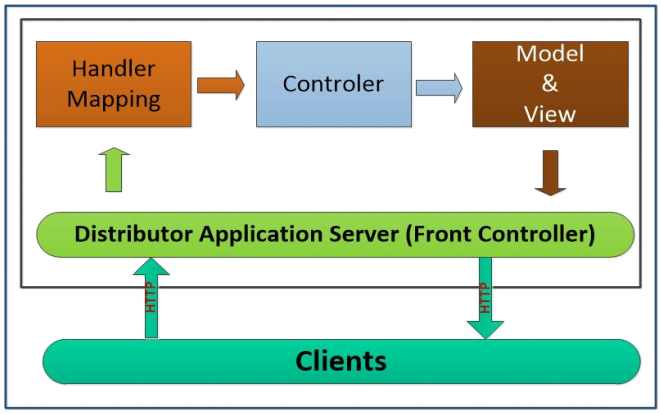

Figure 4. *The architecture of Spring MVC RestFul web service [24]*

The architecture in Figure 4. meets the requests coming from HTTP on the front controller and distributes to the related places. In the process of distribution, the requests are transferred to the related controllers after handler matching. Each request and response are conducted over HTTP. The web services developed for the M2M platform are regulated in accordance with this structure. Communication scenario for all web services developed in this direction is shown in Figure 5.

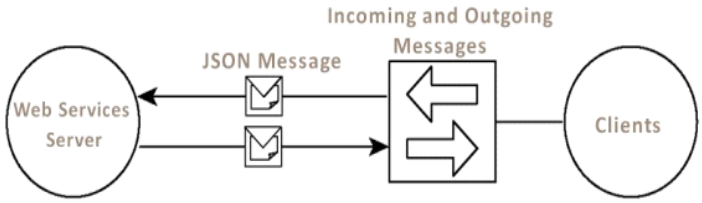

Figure 5. *Web service communication scenario*

As is shown in Figure 5. the messages received from the clients are transferred to web service server in JSON format. Web service server meets the requests, interprets them, and matches with the web service required. The database communication scenario occurred in this situation is shown in Figure 6.

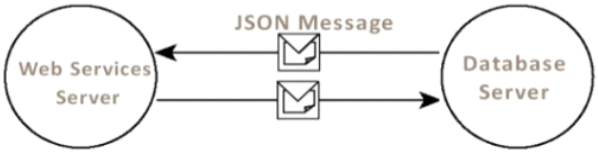

Figure 6. *The scenario of web service and database communication*

As it is obvious in Figure 6. the database is contacted according to the type of the request received by the web service server. The writing or reading transaction is done in this direction. If a task is requested to be done, the business interfaces start and the tasks are done on this layer. The web service scenario of the M2M platform developed is shown in Figure 7.

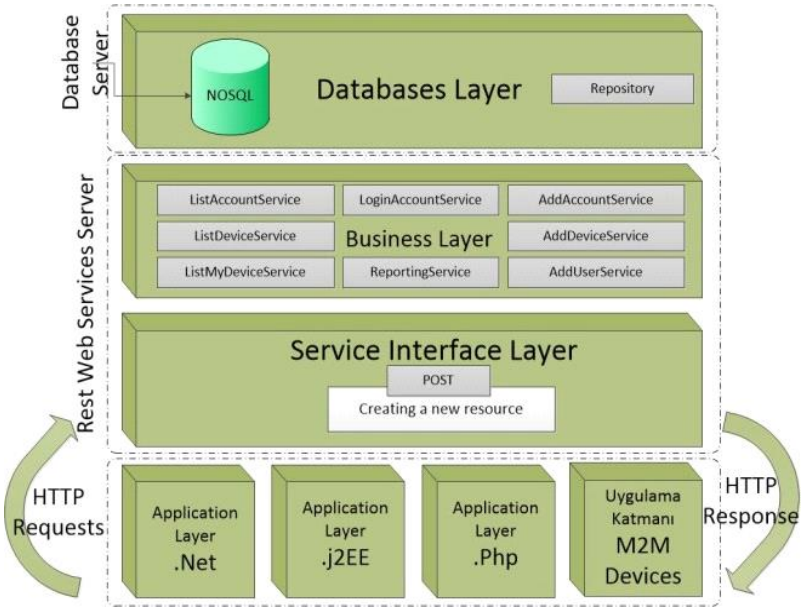

Figure 7. *Web service task scenario*

In the development of M2M platform, the task scenario in Figure 7. is used. According to the figure, any application or M2M device developed by using any programming language sends a request over HTTP to service interface layer. The service interface layer which receives service request with the POST method forms a new source and send the information to the task layer. The task layer matches the information with the related web service, afterward, the task is fulfilled. For instance; many transactions including user adding-listing, device adding-listing, listing-reporting of the device information that belongs to the user, listing-reporting of the device information are done in the business interface layer. Additionally, the writing-reading transactions on the database are also realized by the task layer. In this way of use, the new task processes can be used through integrating the task layer. The stage of conversion of scenario, architecture, and technologies into software is shown in Figure 8.

As it can be seen in Figure 8. the defining of requests and responses is done in the number 1 software packages. On the other hand, the defining of the controllers is done in the controllers package shown in number 3. While controllers are preparing the responses, they may send a request to database for writing and reading transactions. In order not to create complexities in this situation, business interface is developed. Additionally, a layered structure is developed in the software. In the number 2 shown in Figure 8., the tasks that are requested or needed to be fulfilled by the controllers are done. The database transactions are done in the package defined in number 5. Models and appearances are related to the issue of how the responses will be prepared and presented. Along with models and appearances in the packages number 4 and 7, the transactions of data conversion tasks are given. The package number 6 is the one in which error identifications are done. When an error occurred in the M2M platform the error messages and status messages that will be presented to the user are kept in this package. The standard status messages of Spring structure which are used in the development of the M2M platform are below [25];

- 1XX - Informational
- 2XX - Success
- 3XX - Redirection
- 4XX – Client error
- 5XX – Server error

The requests are able to be received with GET or POST commands while the M2M platform is being developed. However, in terms of reliability and continuity GET causes some breakdowns. GET provides an ease of use in small scale applications where data flow is low. Since there is a constant data flow from the clients on the M2M platform; in all transactions, Post is used. The platform proved to be successfully working when each POSTed request turned "200 OK" status info.

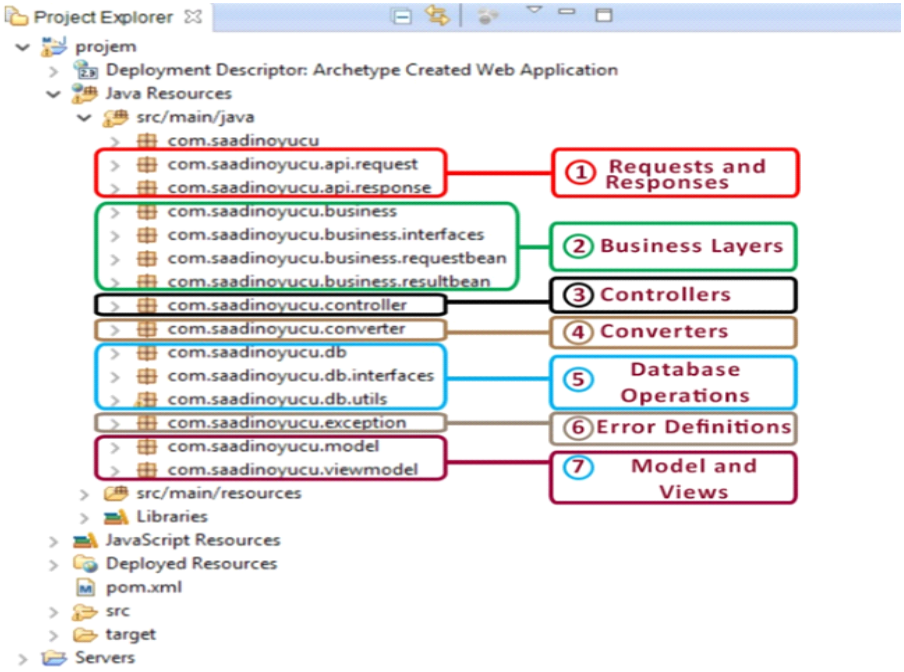

Figure 8. *The conversion of RestFul web service architecture into software*

3.3.4. Security of service

The transactions carried out over URL are more insecure compared to other software systems. The reason behind this is that they are easily visible. Especially GET method's doing transactions overtly over URL makes security processes more difficult. However, if GET method is not used at all in a system, decreases the security gap. In Rest architecture transactions are generally done by the client, therefore server is not overused. When OneM2M's security studies are examined, it will be realized that such practices of

normal software approaches as starting a session and keeping session information on the server are not used. Instead of these approaches checking is done through generating tokens.
In the M2M platform developed session and user checks are done through tokens. The system works as each registered user sends login requests using username and password. Next, web service layer checks the user in the database. If the user is registered to the system, a unique and valid-for-one-hour token and user ID information are sent to the user as a response. If the user wants to make any transaction he has to submit user ID and token information to the web service layer. In addition, the token-generating process is done for M2M devices too. By this way, the security level for M2M devices is aimed to be increased. The security practices necessary for M2M platform is discussed in another study.

3.3.5. Testing the services

Google Chrome web browser provides its users the opportunity to test RestFul web services through the extensions. These extensions are Postman and Advanced Rest Client. To test the system, both extensions are used. Table 3. shows device adding service process testing done through Advanced Rest Client.

The request will be sent to the service address shown in Table 3. As it is clear that request and response formats are JSON, and when the user sends request the necessary information for adding on M2M device software is given to the user when the request is sent. The users of the devices can be identified by this information. Additionally, through generating token information for each device, the security is enhanced. During testing, the possible errors can easily be understood through status checks. Web service tests are conducted for all developed services and the successful results are obtained.

Table 3. *User device adding service process*

Service Address	http://localhost:8080/saadin/service/device/AddDevice
Sample Request	{ "deviceName " : "arduino 1", "token" : "3e500d06-6c53-444a-8e81-18abc99fb34f", "userId": "5507d0b8510dccbfbb9b9fc0" }
Sample Response	{ exceptionCode: 0 hasException: false deviceId: "792a1d92-4924-4644-9e39-8ee2fccdf9d4" deviceName: "arduino 1" userId: "5507d0b8510dccbfbb9b9fc0" validatormessage: null }
Service Description	"Token" sets the session length in the system. "UserID" is the user ID information. DeviceId as user response, DeviceName and are UserId information.

3.4. Preparing M2M Device API's

In this study, Arduino card is decided to be used to set an example. Additionally, the use of a wired IP network technology made our job easier because of its being low cost. Two different suggestions emerged for designing the system to use IP network technology. First one is server streaming on Arduino card, and the second option is creating a client on Arduino card. The first method has some advantages based on the area of use. But the biggest deficiency is that the requests and queries sent are blocked by a firewall. Additionally, it is necessary for Arduino server card that a port should be routed to Arduino card over a modem or router. This way of use is both complicated and uneconomic in terms of IP distribution. The second method which is making a client stream through Arduino card is both economic in terms of IP distribution and its use is easier. For this reason, an API which works as the client structure on Arduino cards was developed. The M2M device client structure which is developed by using Arduino Uno transferred to the network through Arduino Ethernet Shield. To set an example, LM35 heat sensor and DHT11 heat-moisture sensor were used on Arduino. For the Arduino system, which is physically connected, the code to operate properly the necessary code group should be created. At this point, the API's which are saved in the M2M platform is very useful. The user downloads the necessary API's and loads them to Arduino card. After the necessary changes are done on API's, Arduino communicates with

M2M platform and performs its tasks. In the design of API, studies are conducted to send the data in JSON format and the successful results are achieved. The data that is sent over Arduino as JSON is accepted to the service layer as JSON, and written on the database as JSON. In this way, data transformation at any place is not needed.

3.5. Preparing Interfaces

M2M platform needs an interface to interact with the user. Nowadays most of the desktop applications have been replaced with web-based applications. For this reason, a web-based application that is able to interact with the user is developed for the M2M platform. The first step of the development is creating the interface framework. The method has come with HTML 5 and enabled defining separate titles for each transaction is successfully applied to the M2M platform developed. Each title is checked with CSS and visual features are added. In order for the graphics to be presented in motion, JQUERY libraries are used in the platform. It is highly important to follow the standards and to consider color harmony for the sake of usability. In this study, the studies are conducted taking color harmony into consideration.

Desktops, laptops, tablet PCs and smartphones are among the main devices that may interact with user. The screen resolution and sizes of these devices vary. For this reason, the interfaces should have a structure that can properly work in any system. The M2M platform which is developed by taking this need into consideration can work on any device properly. Thus making software specific to each system for smartphones or tablet PCs is not necessary. Instead, an interface with a sensitive design that can properly work on internet browser is presented to the user. In order for sending requests to web service layer developing a Rest client is required. This necessity is met through a PHP-based web page.

3.5.1. Testing interfaces

In order to test the sensitive web interfaces with the M2M platform developed; the web tools came with Google Chrome 12 are used. To set an example, the appearances on iPhone 6, Samsung S4 and HTC One smartphones are tested, and their screenshots are shown in Figure 9. while the screenshot of iPad 3 is shown in Figure 10.

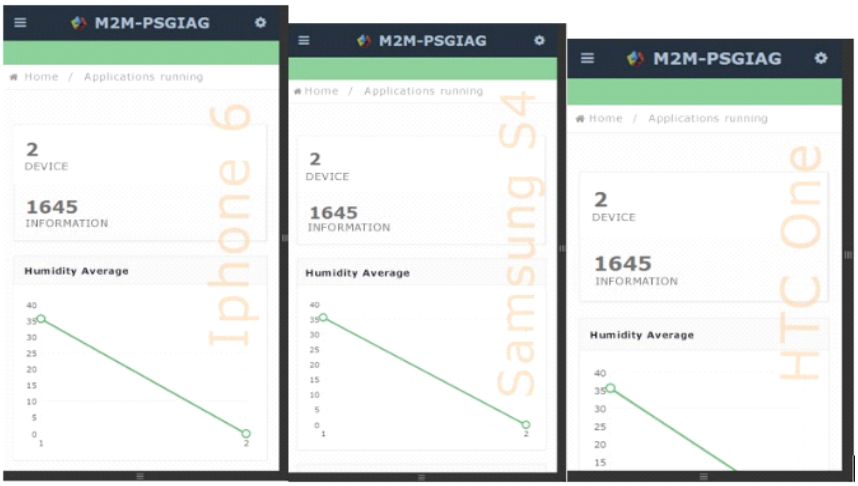

Figure 9. *The appearance of the platform in smartphones*

As it is shown in Figure 9. and 10. The M2M platform is developed to have different appearances in different devices. As a consequence of this feature, the usability is enhanced. Additionally, in the tests conducted with different web browsers, it has been found out that the platform works properly. Thus the need to develop a different application specific to each web browser and screen size is eliminated.

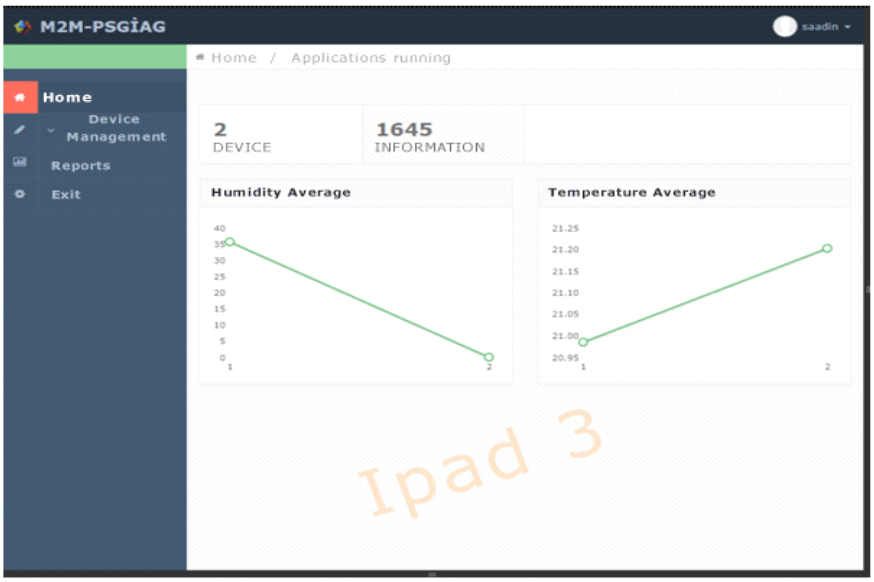

Figure 10. *The appearance of the platform in Ipad 3*

4. THE USE CASE OF THE M2M PLATFORM PERFORMANCE ANALYSIS

Every user who intends to login the M2M platform needs to register first. After registration, the user can login the platform from the login page. The users who will use the system the first time can add a new M2M device using the add device menu. One user can add more than one device to the platform. After the device is added the previously prepared API's are presented to the users to program the M2M devices. The user downloads the API that is compatible with his device and programs his M2M device. Now the M2M device is in a condition that can communicate with the platform. Even if the user is not online in the system, M2M devices continue to communicate with M2M platform.

When the user logs in, the welcome screen is displayed. In this screen, the information about how many devices are added to the platform by the user and how much data is received from the devices can be tracked (Figure 10). Since the data received from M2M devices are continuously saved in the database, the data received from the M2M devices can be tracked back. Additionally, the data received from M2M devices which are added to the platform can be seen in charts (Figure 11). Reporting is also done for the data received from M2M devices. Reports are prepared specifically to the M2M device added to the platform. Data can be tracked on a daily, weekly, or monthly basis. Certain analysis can also be done on the data (Figure 12).

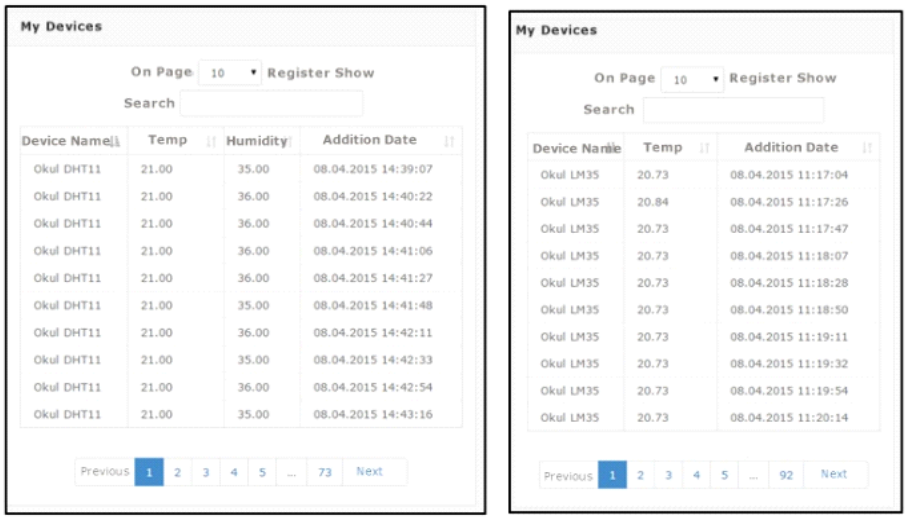

Figure 11. *Tracking the data received from M2M device*

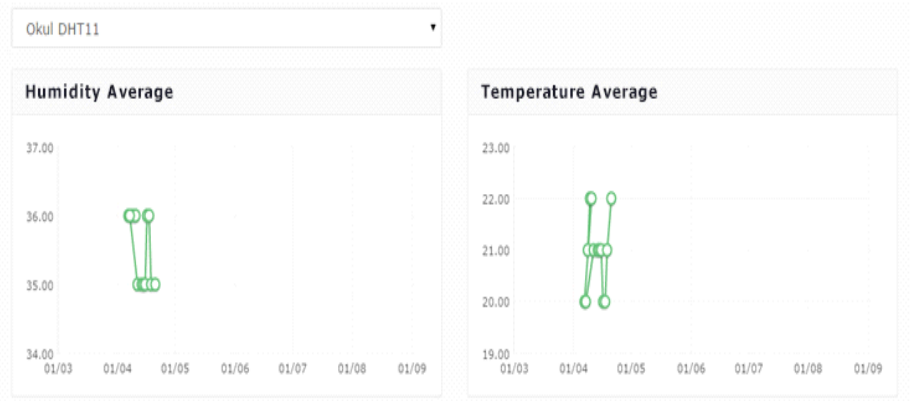

Figure 12. *Reporting the data received from M2M device*

M2M platform is developed in a way to give successful results in high reading and writing transactions. The speed performance of M2M platform in tests conducted with different client tools is available in Table 4.

M2M platform is developed in a desktop which had a 4-core processor and a 4 GB RAM through running Tomcat 7.0, XAMMP PHP server and MongoDB database at the same time. The results obtained from queries done by Google Rest clients are shown in Table 4. As in Table 4. requests and responses are measured in millisecond type. M2M platform shows a high performance in situations where more than one data record is requested. Each layer of the M2M platform can be arranged to operate on a different computer or server. In this situation, the speed of the connection between these computers or servers can affect the overall performance of the system.

Table 4. *M2M platform speed performance*

Request Performance		Response Performance			
Request Type	Submission Time				
User Register	12 ms	Device Lists		Device Information	
Logon	10 ms	Piece	Submission Time	Piece	Submission Time
Add Device	20 ms	1	9 ms	10	8 ms
1 Device Data	26 ms	2	9 ms	570	16 ms
10 Device Data	26 ms	5	9 ms	830	17 ms
100 Device Data	27 ms	10	10 ms	1000	21 ms
1000 Device Data	26 ms	100	10 ms	10000	22 ms

5. RESULTS AND RECOMMENDATIONS

The current technologies have been trying to realize communication between machines and objects with each other. The sensor technologies have been constantly renewed and improved. This situation led to M2M applications spread and use in all fields. In order for M2M applications to interact with the user, there is a need for an M2M platform. Nowadays many M2M platforms have been developed and presented to the users without meeting the requirements of standards. Under the scope of this study, an M2M platform which is compatible with the standards provided by OneM2M has been developed. With the M2M platform developed, more than one user can add more than one M2M device to the platform at the same time. The architecture of the M2M platform is developed using the approaches of the institutions, which globally produce standards on M2M platforms. In this architecture with the features added to the business interface, the abilities of the M2M platform are enhanced.

In the M2M platform, a non-relational database is used. In this way, the scalability and performance of the data system of the M2M platform are enhanced. Moreover, this platform is developed in accordance with the RestFul web service architecture standards. As a result of this, the M2M platform is able to respond the multiple reading-writing requests without delay.

The security of M2M platform is a highly important issue. A security gap occurs since RestFul architecture communicates over HTTP. However in the M2M platform developed, the web services' security is increased through a token-based ID check. Thanks to this under the principle of authorized and unauthorized requests; general system security is enabled. Another factor that increases the security is the use of only POST method instead of GET method. With the help of the layered structure, the intervention in the possible security gaps that may occur in future are made easier.

The M2M platform is developed through a user-friendly design. Therefore the need to prepare different designs for each mobile system is eliminated. The user is able to connect to the system from any system in which he can use the internet connection and web browser.

Only JSON data structure format is used in the M2M platform developed. The data received from M2M devices are stored in the database as a JSON document. In this case, the format of the data is not changed and the time is not wasted with transformation transactions. This way of use decreases the costs and increases the performance and the scalability.

CONFLICTS OF INTEREST

No conflict of interest was declared by the authors.

REFERENCES

[1] Polat, H., and Oyucu, S., "Token-based authentication method for M2M platforms" Turkish Journal of Electrical Engineering & Computer Sciences, 25(4): 2956-2967, (2017).

[2] Yang, Y., Ye, H., and Fei, S., "Design of communication interface for M2M-based positioning and monitoring system", International Conference on Electronics, Communications and Control (ICECC), 2624-2627, (2011).

[3] Chao, M., He, J., and Chen, H. H., "Uncoordinated coexisting IEEE 802.15. 4 networks for machine to machine communications", Peer-to-Peer Networking and Applications, 7(3): 274-284, (2014).

[4] Steinheimer, M., and Ulrich T., "Decentralised system architecture for autonomous and cooperative M2M application service provision", IEEE International Conference on Smart Grid and Smart Cities (ICSGSC), 312-317, (2017).

[5] Castro, M., Jara, A. J., and Skarmeta, A. F., "An analysis of M2M platforms: Challenges and opportunities for the internet of things", Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 757-762, (2012).

[6] Meddeb, M., Alay, M. B., Monteil, T., Dhraief, A., and Drira, K., "M2M platform with autonomic device management service" Procedia Computer Science, 32(1): 1063-1070, (2014).

[7] Kim, E. J., and Youm, S., "Machine-to-machine platform architecture for horizontal service integration", EURASIP Journal on Wireless Communications and Networking, 1(1):1-9, (2013).

[8] Song, J., Kunz, A., Schmidt, M., and Szczytowski, P,. "Connecting and managing M2M devices in the future internet", Mobile Networks and Applications, 19(1): 4-17, (2014).

[9] Takatsuka, H., Saiki, S., Matsumoto, S., and Nakamura, M., "Design and implementation of rule-based framework for context-aware services with web services", 16th International Conference on Information Integration and Web-based Applications & Services, 233-242, (2014).

[10] Alaya, M. B., Banouar, Y., Monteil, T., Chassot, C., and Drira, K., "OM2M: Extensible ETSI-compliant M2M service platform with self-configuration capability", Procedia Computer Science,

32(1): 1079-1086, (2014).

[11] Wollschlaeger, M., Sauter, T., and Jasperneite, J., "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0", IEEE Industrial Electronics Magazine, 11(1): 17-27, (2017).

[12] Marin-Perianu, M., Meratnia, N., Havinga, P., Souza, L. M. S., Mller, J., Spie, P., and Stromberg, G., "Decentralized enterprise systems: A multiplatform wireless sensor network approach" Wireless Communications, 14(6): 57-66, (2007).

[13] Zhaozong M., Zhipeng W., and John G. "A collaboration-oriented M2M messaging mechanism for the collaborative automation between machines in future industrial networks", Sensors, 17(11): 2694, (2017).

[14] Pencheva, E., and Atanasov, I., "Engineering of web services for internet of things applications" Information Systems Frontiers, 18(2): 1-16, (2016).

[15] Internet: "The world's first global standards for M2M", http://www.onem2m.org/newsevents/news/53-the-rise-of-the-achines-world-s-first-global-standards-for-m2m-eployment, (2017).

[16] OneM2M Technical Report, Document Name: "Architecture Analysis - Part 1: Analysis of architectures proposed for consideration by oneM2M", Document Number : oneM2M-TR-0002-Architecture_Analysis_Part_1-V-0.2.0, (2013).

[17] Yishan, L., and Sathiamoorthy, M., "A performance comparison of SQL and NoSQL databases", IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), 15-19, (2013).

[18] Han, J., Haihong, E., Le, G., and Du, J., "Survey on NoSQL database", 6th International Conference on Pervasive Computing and Applications, 363-366, (2013).

[19] Cicek, A. N., "RestFul web services with e-health systems implementation", Master Thesis, TOBB Economics and Technology University, Graduate School Of Natural And Applied Sciences, Ankara, (2009).

[20] Erl, T., "Service-oriented architecture: Concepts, technology, and design", Pearson Education India, 10-89, (2005).

[21] Fielding, R. T., "Architectural styles and the design of network-based software architectures", Doctoral dissertation, University of California, Irvine, (2000).

[22] Liang, X., Chi, K., and Huang, M,. "Design and implement for internet taxpaying system based on spring MVC", 4th International Conference on Computer Science and Network Technology (ICCSNT), 1: 425-430, (2015).

[23] Internet:Spring, http://projects.spring.io/spring-framework/, (2017).

[24] Zhang, D., Wei, Z., and Yang, Y., "Research on lightweight MVC framework based on spring MVC and mybatis", Sixth International Symposium on Computational Intelligence and Design (ISCID), 350-353, (2013).

[25] Hindriyanto, P.,Dody, S., Ramos, S., and Charitas F. "The application of restful web service and JSON for poultry farm monitoring system", Journal of Electrical Engineering and Computer Sciences, 1:1, (2016).