Design and Development of An ASP.NET Based ITM Data Center To Support Open Data

Rahman, Md. Rahatur[1]; Al Forhad, Md. Abdullah[1]; Ahmed, Salah Uddin[2]; Akhter, Md. Shamim[1]

[1]International University of Business, Agriculture and Technology, Dhaka

[2]Department of Business, Marketing and Law - University of South-Eastern Norway

This is an Accepted Manuscript of an article published by IEEE Online in *2021 International Conference on Information Technology (ICIT)* on July 21, 2021, available online:
https://doi.org/10.1109/ICIT52682.2021.9491756

Rahman, M. R., Al Forhad, M. A., Ahmed, S. U., & Akhter, S. (2021, July 14-15). Design and Development of An ASP.NET Based ITM Data Center To Support Open Data [Conference presentation]. *2021 International Conference on Information Technology (ICIT).* Amman. https://doi.org/10.1109/ICIT52682.2021.9491756

Design and Development of An ASP.NET Based ITM Data Center To Support Open Data

Md. Rahatur Rahman, Md. Abdullah Al Forhad
AISIP Lab, Dept. of Computer Science and Engineering,
International University of Business Agriculture and Technology
Dhaka, Bangladesh
rahatur.rahman@gmail.com

Salah U. Ahmed
University of South-Eastern Norway (USN), Dept. of Business, Marketing and Law, Campus Ringerike,
Norway
Salah.Ahmed@usn.no

Shamim Akhter
AISIP Lab, Dept. of Computer Science and Engineering
International University of Business Agriculture and Technology
Dhaka, Bangladesh
shamimakhter@gmail.com
https://orcid.org/0000-0003-1408-9133

***Abstract*—Intelligent Traffic Management (ITM) helps to solve real-time traffic problems and guides efficient and effective routes to reach a destination. It aggregates information from various sensors located in different places of roads and in vehicles to collect different kinds of data about vehicles, weather, roads, and traffic, etc. These data are filtered and processed to generate results from which ITM generates appropriate communication-related decisions. The full ITM must be able to cooperate by allowing communication with and among vehicles and/or IoT devices. It creates pressure on the network, requiring high data transmission bandwidth, and demands short response time and latency for our time-sensitive traffic applications. Besides these, massive amounts of data also demand faster processing and secure storage. In this context, a data center is deemed an ideal companion for ITM which will be used for storage, processing, and transmission of data and results back to different clients. In this article, we are going to present a data center that is specially built for ITM. Our designed data center uses WebSocket-based bi-directional communication, load balance, fault tolerable module, data replicability, and provides road-vehicle-traffic-related web services and distributes Open Data with API supports.**

Keywords— Data Center, Intelligent Traffic Management (ITM), WebSocket, Open Data, IoT Devices

I. INTRODUCTION

We proposed and implemented a low-cost, flexible, internet-based, and bidirectional Intelligent Traffic Management System-ITMS [1][2]. It solves traffic-related optimal routing problems. ITMS takes weather, road, and vehicle-related data from a road segment on an hourly basis and dynamically calculates the weight of this road segment using the Deep-Neuro-Fuzzy model [3]. Besides, the Deep-Neuro-Fuzzy model was implemented by a simulation-based environment with the SUMO tool [4][5] and proved the workability of the model. The next step is to implement the ITMS in a real testbed with real traffic scenarios.

The complete ITMS requires a server or data center to support WebSocket-based bidirectional client-server communication, secure storage of data, low latency, higher computational power, reliable connection, service-oriented platform, and web-based representation. Besides, it supports APIs to offer public or private Open Data repositories. In this regard, an ASP.NET-based ITM data center is designed and developed. In this paper, we are going to describe the implementation of the data center with the necessary modules. Section II presents some information about the data centers in Bangladesh and related works with ITMS applications. Section III describes the configuration of the servers and the integration of different modules with servers. Section IV describes the activities of the data center including data storage to service. Section V concludes and highlights future researches.

II. RELATED WORKS

The Uptime Institute standard or the American standard TIA-942 categorizes data centers into four levels or Tiers from I to IV according to redundancy, service availability, power consumption, and reliability. In Bangladesh, XeonBD [10] maintains a data center for providing web hosting services. A Tier IV National Data Center has been established at the 'High Tech Park' in Kaliakoir, Gazipur by Zhong Xing Telecommunication Equipment (ZTE) Corporation [11]. This data center capacity will be two petabytes. DevoTech[12] and DhakaColo[13] provide data centers for web hosting and data storage, however, they did not define any standards or levels. Although currently, we are maintaining Small Business Servers as a Tier I types data center, however, our design represents the Tier II category. We use privately a dedicated low-cost data center for the traffic management system and we have a plan to connect the resources with [11].

In [14] authors proposed a cloud computing-based framework to store, manage, process, and analyze traffic-related data. Zhejiang city in China uses an Apache Hadoop cluster as a data center with 22 Xeon processor E5 series servers and 198 terabytes of storage space [15]. Thus, it is common that road traffic-related data are stored in a cloud-based server or multiple computers-based data centers. We have already implemented our low-cost flexible and reliable ITMS [1-5] into public cloud-based architecture in [6]. However, to include the public services or provide the Open Data framework needs private storage facilities like a data center with limited budget commodity hardware. We assume that the Open Data initiative removes technical barriers and acquisition barriers for others to use data without any extra cost, and in this way opens up possibilities to create meaningful information services in the involved community. The major challenges in this research are as follows:

- Take away all separate ITMS modules from the simulation environment, convert them as API, and then port them into a real centralized data center.
- Connect the modules with their existing interoperability.
- Implement Open Data Framework and distribute APIs.

III. Intelligent Traffic Management (ITM) System

This section describes the ITM structure that will be implemented on the server of the data center. The ITM algorithm scraps data from the predefined external data sources (like website, RSS feed, web service, etc.). In addition, it feeds the data from edge sensors to the database. Deep-Neuro-Fuzzy Model evaluates the weight of a road segment using the stored data. Swarm Intelligence Algorithm namely Ant Colony Optimization(ACO) uses the weights and decides the shortest or collision-free path from the source to the destination for a particular vehicle. Fig.1 describes the full structure of the ITM [17][18][19][20].

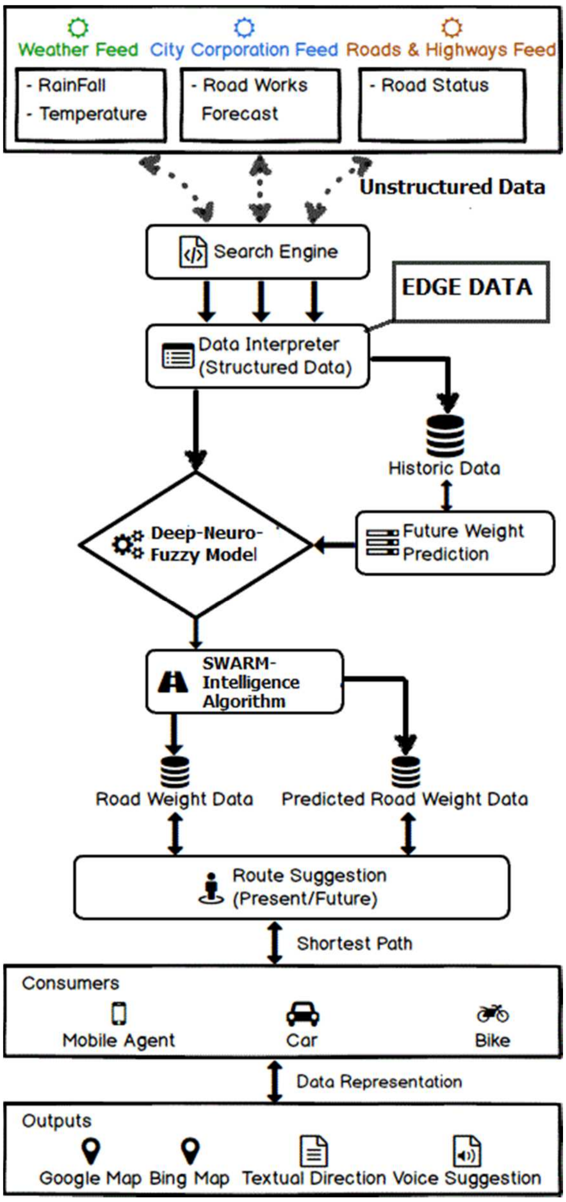

Fig.1. Workflow model of the ITM

IV. Implementation of the Data Center

This section describes the system requirements and the working procedures of the data center and its software that stores and processes the raw data collected by the field sensors. Based on the processed data the system provides web services that can be consumed by third-party applications through web services.

A. Implementation of IoT devices

Internet of Things (IoT) is a network, where remote edges are interconnected and able to communicate with each other (sensor-enabled communication). IoT in traffic systems needs to collect and integrate real-time traffic information, process, and analysis them automatically, and make the system more self-reliable and intelligent. We design and implement a secure IoT-based framework for real-time traffic-related information acquisition and monitoring architecture, and to implement it with several remote agents utilizing wireless communication (GSM/GPRS/Bluetooth). Each agent is an integrated embedded device with different sensors and controllers namely DHT11, Raindrop's sensor, Arduino microcontroller, and GSM/GPRS/GPS/Bluetooth enabled modules. Our designed devices are securely interconnected with an authenticated data center using GSM/GPRS-based HTTP protocol, and able to collect or exchange encrypted data among themselves through GSM/GPRS/Bluetooth wireless communication [6]. Devices can collect specific data with a degree of autonomy and interact with the data center without human intervention. Besides, the devices have a fault tolerance system as they are capable to communicate (send and receive data) with the data center even with the failure of the GSM signal. They do this by connecting another nearby device or authenticated mobile phone (GSM enabled and has signal) using Bluetooth communication and request him (device/phone) to communicate with the data center. Besides, AES 128 CBC Encryption and/or decryption algorithms are used to secure the data transmission.

The GPS-enabled agent is referred to as an edge [9] that broadcasts its current location in terms of latitude and longitude. The edge is also equipped with GSM/GPRS/GPS/Bluetooth SIM808 Shield that is accomplished to handle the web request/response over the HTTP. At first, all data are being gathered as plain text then the HTTP POST method is being used to send the data to the server. The type of the body of the request is indicated by the Content-Type header. The edge device needs the following parameters for the request,

mac1, temp1, hum1, rain1, lon1, lat1, vel1

Here, mac1 is being used to store and send the MAC address of each device which is being used to identify the physical edges. Because it is necessary for us to uniquely identify every edge for a valid and accurate result. temp1 is being used to store and send Temperature data in degree Celsius. hum1 is being used to store and send Humidity data in percent (%). rain1 is being used to store and send rail fall data in a range of 0 to 1024 in which 0 means heavy rain and 1024 means no rain. lon1 is being used to store and send longitude information of a location. lat1 is being used to store and send latitude information of a location. vel1 is being used to store and send the velocity information in which the embedded device is being moved. The action performed by the POST method might not result in a resource that can be identified by a URI. In this case, either 200 (OK) or 406 Not Acceptable is the appropriate response status, depending on whether or not the response includes an entity that describes the result. And the following is a general connection request example from the edge:

POST //api/Telemetry HTTP/1.1

Host: rahat651-001-site3.itempurl.com

User-Agent: Arduino

Content-Type: application/x-www-form-urlencoded

Content-Length: 99

mac1=5244-5874-1254-9423&temp1=33.00&hum1=79.00&rain1=741&lon1=90.412521&lat1=23.810331&vel1=1.12

Fig. 2 presents the full procedure to communicate between an edge and the server.

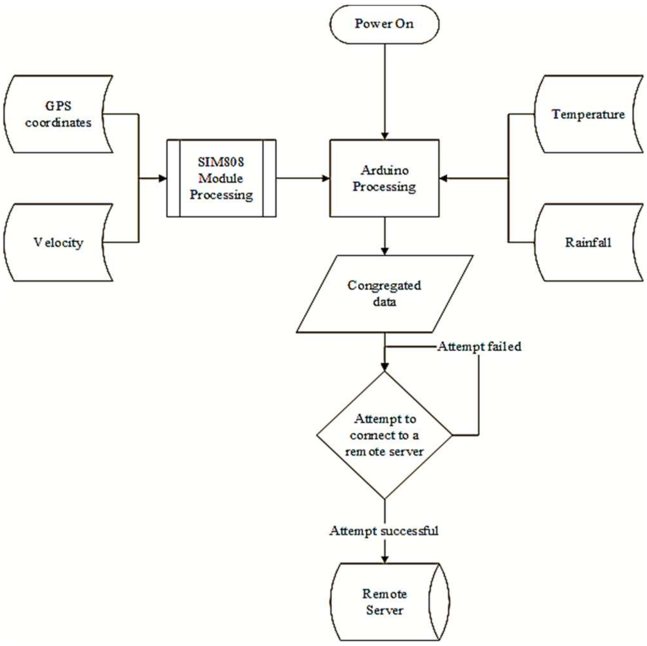

Fig.2. Communication between edge to server

TABLE I: Specification of the Servers

Web Server	Database Server
CPU: Intel Xeon 2 Core or equivalent RAM: 8GB Storage: 50 GB Bandwidth: 2TB Web Server: IIS 8.5 Operating System: Windows Server 2012 or higher Firewall: Standard Compliance: SSAE 16 SOC 2 Type 2 Compliance Encryptions: TLS 1.2 Certificate: SSL	CPU: Intel Xeon 2 Core or equivalent RAM: 4GB Storage: 100 GB Bandwidth: 2TB Operating System: Windows Server 2012 or higher Database: MS SQL Server 2012R2 or higher Firewall: Standard Compliance: SSAE 16 SOC 2 Type 2 Compliance Encryptions: TL

B. Web Application and Services

Intelligent Traffic Management (ITM) acts as the interface of the data center. The key responsibility of the data center is to receive data from various devices running on the field, store the raw data, process them, and expose web services for the third-party application to consume the processed data.

The ITM system is implemented based on the following technologies: Language: C#, JavaScript, HTML, CSS, Python, LINQ Frameworks, Asp.net MVC 5, Dotnet 4.5.1, Entity Framework, Identity Framework. Libraries: Google Map, SignalR, jQuery, Bootstrap.

C. Configuration of the Servers

Table I presents the Web Server and Database Server specifications.

D. Implementation of WebSocket

WebSocket is an application protocol that supports bidirectional communication over TCP protocol port 80. WebSocket API is being standardized by the W3C and usually use to implement real-time, interactive, and less latency demanded applications. The server/host is compatible with HTML5, WebSocket, and provides services to the clients. Here clients are the customers who need public services or data services or API executions in the server. At first, the client sends a request to the server for a WebSocket connection. The server allows an authenticated client and authorizes to access its resources. The server maintains a session for each client and the WebSocket connection remains open for the whole session until any endpoint request is released. Data are transferred from the server to the client or vice versa with no preceding request during an active session. In our implementation, the server also hosts the business analytical parts of our web application, thus it can handle multiple clients at the same time. The server is built as a web application and runs on top of the open-source Microsoft Asp.net MVC platform. For the initial WebSocket connection, SignalR based codes are used. The code snippet below demonstrates the initialization on the client-side:

```
<script src="~/Scripts/jquery.signalR-2.2.0.min.js"></script>
<script src="~/signalr/hubs"></script>
<script
src="~/Scripts/signalRStateManager.js"></script>
<script type="text/javascript">
// to get user identity from signalr hub

$.connection.hub.qs = { uname:
'@ViewContext.HttpContext.User.Identity.Name' };

$(function()
{
        $.connection.Event.client.method1 = method1;
        $.connection.Event.client.method2 = method2;
});

function method1(posData)
{
}

function method2(posData)
{
}
</script>
```

The above script establishes the connection between the client and the server and also lets the server identify the requested method from the client request. The server identifies the method (method1) and pushes a JSON object into it.

```
public static void BroadcastEvent(Event model)
{
        var jsonObj = new
JavaScriptSerializer().Serialize(new { model.Id,
model.Name, model.DateCreated });
```

```
        var signalrHub =
GlobalHost.ConnectionManager.GetHubContext<EventHub>
();
        signalrHub.Clients.All.method1(jsonObj);
}
```

In a real-world scenario, WebSocket may not be supported by a group of clients. To handle this, we need to have a fallback mechanism so that our services can run without any interruption. Thus, a backup long pooling strategy is also available to make a successful connection between client and server.

E. The Data Center

According to Fig.3, IIS-Internet Information Service is the back-end web server and maintains secure channels to communicate with the remote/field sensors. Raw data is processed by the system and kept separately for serving through the web services. The web servers are balanced with a network load balancer to distribute the loads fairly between the servers (Fig.5). This procedure helps a failsafe mechanism when one of the servers is offline. MS SQL database servers are used to store both raw and processed data as well as historical and system configuration. Fig.4 shows a screenshot of some processed data using [3]. We use a replicated database server to store the date-wise offsite database to overcome the data loss during the disaster scenarios (Fig. 6).

F. Integrating SWARM-Intelligence Algorithm for Routing

We expose a new multi-agent-based Ant Colony Optimization (ACO) system into the SUMO simulator tool to make the ITM more robust and self-adaptive [4][5]. Thus, we modify and adjust the ACO according to the ITM and SUMO implementation. In our implementation, we consider road segments as links and road speeds are trails or pheromone levels. We apply Grid Search on different internal parameters namely pheromone density, pheromone trail, visibility, and control parameters namely α and β of the ACO to find their optimum setting to route more vehicles from the source to a destination within a certain period. Thereafter, a comparative study is done on existing CHWrapper, A*, and Dijkstra route-finding algorithms and our multi-agent-based ACO system. ACO performs best among all the others existing algorithms in SUMO simulated platform. Thus, the Dijkstra shortest path algorithm is replaced by the ACO model on the server-side.

V. ACTIVITIES OF THE DATA CENTER

A. Data Collections from Remote Sensors and Storage

The core purpose of the remote edge or vehicle is to collect the telemetry data and transmit it to the TMS server in a regular interval (26 seconds). The edges are comprised of some customized software and hardware sensors. Arduino is an open-source electronics platform or board supports as a processing unit for the device. Software is used to program it. The remote edges collect and transmit data including MAC Address, Surface Temperature, Relative Humidity, Rainfall, Velocity, Latitude, Longitude to the servers. Fig. 8 presents some sample data transmitted by two (2) different remote agents (two different mac addresses) which are stored on the data center.

B. Public Services as Open Data

The ITM application processes the following data and offers the result as web services:

- The calculated weight of an area/road
- The average speed of vehicles in an area/road
- The surface temperature of an area/road at a given time
- The average surface temperature of an area/road
- The humidity of an area/road at a given time
- Average Humidity of an area/road
- Rain Fall of an area/road at a given time
- Average Rain Fall of an area/road

The data are processed by the back-end system and exposed as Representational state transfer API (RESTful Web Services). This provides interoperability between different systems over the HTTP protocol. Open Data (OData) is an ISO/IEC approved and OASIS standard protocol that defines the building blocks to consume RESTful APIs. The OData V4 standard is followed to transmit the data to the API consumers. Consumers' programs also need to follow the same standard and confirm the continuous communication between the application and the ITM data center. To provide useful data for relevant users, we have designed the data center to provide data in four levels including Dataset, API, Concept, and Information Model(in Fig.9). As with the current version, we have data from sensors as shown in Fig.7 are stored in Datasets, Datasets are stored as a table, and the table can be downloaded as XLSX, CSV, and JSON-stat format. Any user can access data by using API by formulating a query that works as a POST method to a URL. A sample JSON query looks like in Fig. 8. The output formats are CSV (semicolon-separated files) and JSON-stat. Datasets and data services are described using standard model and vocabulary following W3's Data Catalog Vocabulary (DCAT) recommendations which, facilitates the consumption and aggregation of metadata from multiple catalogs. DCAT facilitates interoperability between data catalogs published on the Web and increases the discoverability of datasets and data services. In the future we would like to expand the number and type of sensors, therefore we have planned to include concepts and information models in our data center. Concepts will describe the information of the data, its type, properties, relationships, and operations. Information Model will provide the framework for organizing the contents so that they can be delivered and reused in a variety of innovative ways. Thus, our next aim is to design the Concept and Information Model so that consumers can find the information resources they need quickly and easily. Not only direct consumers but also the providers can able to share our data through their repositories as well as sharing their data in our data center. Therefore, the data center aims to host ITM-related datasets from other providers in the future.

VI. CONCLUSION

An ASP.Net-based data center is successfully implemented for serving ITM services. It supports WebSocket-based bi-directional communication. Thus, the communication overhead for the system is reduced. Real-time data from several IoT sensors are collected and stored

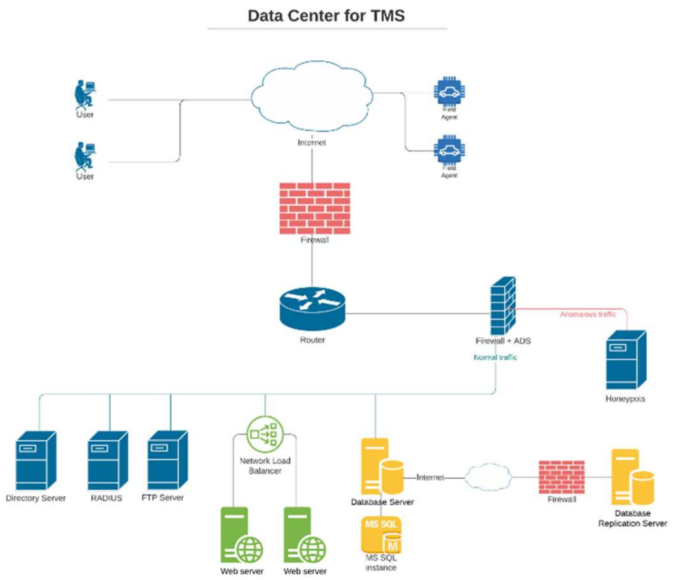

Fig. 3. ITM Data Center

WightID	FromID	ToID	RoadWeight
1	2	3	1000
7	2	6	5
8	3	2	5
9	3	15	5
10	3	4	5
11	4	3	5
12	4	23	5
13	4	5	5
14	15	16	5
15	15	3	5
16	15	19	5
17	16	15	1000
18	16	17	5

Fig. 4. Screenshot of Some Processed Data

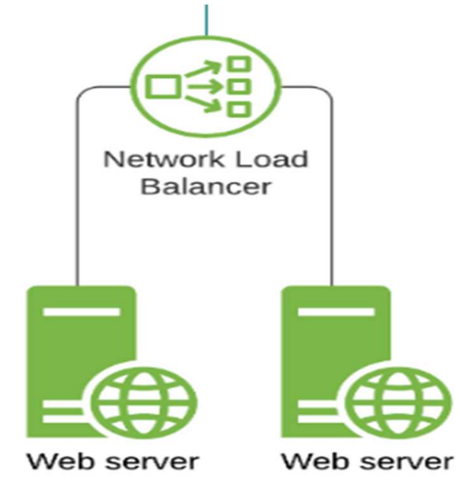

Fig. 5. Web servers and their connectivity

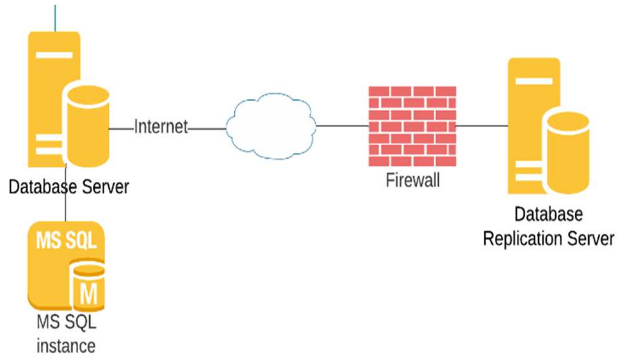

Fig. 6. Database servers and their connectivity

Id	MacAddress	SurfaceTemperature	RelativeHumidity	RainFall	Latitude	Longitude	Velocity	DateCreated	DateUpdated
218	5244-5874-1254-9424	26.000000	66.000000	1023.000000	23.874120	90.390755	0.020000	2019-12-23 10:57:23.087	2019-12-23 10:57:23.087
219	5244-5874-1254-9425	30.000000	49.000000	1005.000000	23.874142	90.390785	0.000000	2019-12-23 10:57:29.697	2019-12-23 10:57:29.697
220	5244-5874-1254-9424	26.000000	66.000000	1023.000000	23.874120	90.390755	0.020000	2019-12-23 10:57:49.207	2019-12-23 10:57:49.207
221	5244-5874-1254-9425	30.000000	49.000000	1005.000000	23.874142	90.390785	0.000000	2019-12-23 10:57:55.720	2019-12-23 10:57:55.720
222	5244-5874-1254-9424	26.000000	66.000000	1023.000000	23.874122	90.390753	0.020000	2019-12-23 10:58:15.080	2019-12-23 10:58:15.080
223	5244-5874-1254-9425	30.000000	49.000000	1005.000000	23.874142	90.390787	0.020000	2019-12-23 10:58:21.817	2019-12-23 10:58:21.817
224	5244-5874-1254-9424	26.000000	66.000000	1023.000000	23.874122	90.390753	0.000000	2019-12-23 10:58:41.127	2019-12-23 10:58:41.127
225	5244-5874-1254-9425	30.000000	49.000000	1005.000000	23.874142	90.390787	0.000000	2019-12-23 10:58:47.723	2019-12-23 10:58:47.723
226	5244-5874-1254-9424	26.000000	66.000000	1023.000000	23.874143	90.390755	8.590000	2019-12-23 10:59:07.190	2019-12-23 10:59:07.190
227	5244-5874-1254-9425	30.000000	49.000000	1005.000000	23.874357	90.390753	20.590...	2019-12-23 10:59:13.817	2019-12-23 10:59:13.817
228	5244-5874-1254-9425	30.000000	49.000000	1009.000000	23.876047	90.390777	26.110...	2019-12-23 10:59:46.380	2019-12-23 10:59:46.380
229	5244-5874-1254-9425	30.000000	47.000000	1010.000000	23.879062	90.390697	30.980...	2019-12-23 11:00:05.847	2019-12-23 11:00:05.847
230	5244-5874-1254-9424	27.000000	65.000000	1023.000000	23.880065	90.390337	15.240...	2019-12-23 11:00:25.240	2019-12-23 11:00:25.240
231	5244-5874-1254-9425	30.000000	46.000000	1013.000000	23.880593	90.391025	10.220...	2019-12-23 11:00:57.787	2019-12-23 11:00:57.787
232	5244-5874-1254-9424	26.000000	64.000000	1023.000000	23.880990	90.390920	18.320...	2019-12-23 11:01:17.397	2019-12-23 11:01:17.397
233	5244-5874-1254-9425	30.000000	47.000000	1012.000000	23.881457	90.390873	10.890...	2019-12-23 11:01:23.867	2019-12-23 11:01:23.867
234	5244-5874-1254-9425	30.000000	48.000000	1013.000000	23.882687	90.390380	27.040...	2019-12-23 11:01:50.130	2019-12-23 11:01:50.130
235	5244-5874-1254-9424	26.000000	65.000000	1023.000000	23.883913	90.389977	41.650...	2019-12-23 11:02:09.460	2019-12-23 11:02:09.460

Fig.7. Remotely Collected Data are Stored in Data Center

```
{  "query": [
   {      "code": "Id",
     "selection": {
       "filter": "item",
       "values": [   "218",     "219",     "220"         ]      }    },

  {
     "code": "MACAddress",
     "selection": {
       "filter": "item",
       "values": [  "5544-5874-1254-9424",    "5544-5874-1254-9425",  "5544-5874-1254-9424" ]
     }
  },
............
"response": {     "format": "json-stat2"   } ],
}
```

Fig .8. Sample JSON query using the API

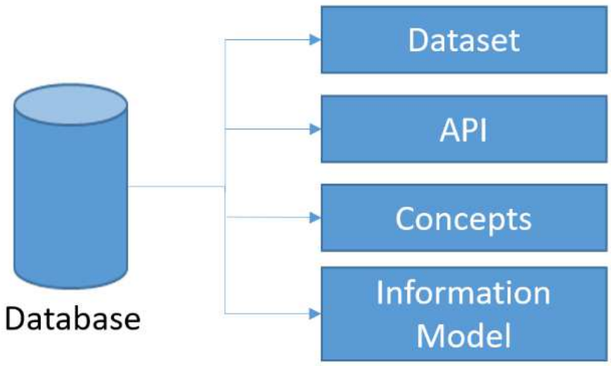

Fig.9. Logical Content Structure of the data center

successfully. Data are distinguished according to the device Mac address and the sensor locations. We use encrypted data sharing between client and server and avoid masquerading and man-in-the-middle attack *(*MITM*)* attacks. The data center is also secured with firewalls and anti-virus tools. In this article, we provided a solution for creating a data center that will support open data architecture also. Many developing countries have a large population and a big portion of the population is the young generation who are open and agile towards new technologies. There is a huge need for information and data services for these people, but limited economic and technical resources act as an obstacle to the development of such services. We believe that locally developed, tailored solutions and services can play an important role in evolving and promoting data-based services that create a huge impact in different government sectors and society. Most of the major cities in the developing world struggle with traffic problems that result in the waste of many useful productive hours for their citizens and business. Our solution provides a framework, which would encourage enthusiastic researchers to develop services that can improve efficiency in the traffic system and reduce existing problems. In the future, this framework can be joined together with larger data centers, and/or extended to provide wider scope and domain areas. We did not consider gateways to connect different networks and their protocol conversion. Fog devices can be implemented and work as gateways. Small tasks like alarming, data processing, data filtering, etc can be services from Fog devices, not the cloud. We are going to implement this in near future.

Acknowledgment

The authors would like to express their sincere gratitude to the Information and Communication Technology (ICT) Division, Government of the People's Republic of Bangladesh for funding the research and the International University of Business Agriculture and Technology (IUBAT), Sector-10, Uttara Model Town, Dhaka, Bangladesh for their necessary supports.

References

[1] M. R. Rahman and S. Akhter. “Real-time bi-directional traffic management support system with GPS and WebSocket”, In Proc. of the 15th IEEE International Conference on Computer and Information Technology (CIT2015), Liverpool. UK,2015.

[2] M. R. Rahman and S. Akhter, “Bi-directional traffic management support system with decision tree-based dynamic routing”, In Proc. of 10th International Conference for Internet Technology and Secured Transactions, ICITST, 2015, London. The United Kingdom.

[3] S.H.Sumit and S. Akhter, “C-Means Clustering and Deep-Neuro-Fuzzy Classification for road weight measurement in traffic management system”, Soft Computing, Springer, Vol 23, Issue 12, June-2019, pp 4329–4340. https://doi.org/10.1007/s00500-018-3086-0.

[4] S. Akhter, N Ahsan, SJS Quaderi, MA Al Forhad, and SH Sumit, “A SUMO Based Simulation Framework for Intelligent Traffic Management System”, Journal of Traffic and Logistics Engineering, Vol. 8, No. 1, pp. 1-5, June 2020. DOI: 10.18178/jtle.8.1.1-5

[5] S. Akhter, N Ahsan, and SJS Quaderi, “Modeling Ant Colony Optimization for Multi-Agent based Intelligent Transportation System”, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 10, No. 10, pp. 277-284, 2019.

[6] M. A. A. Forhad, M. Nadim, M. R. Rahman, and S. Akhter, “Cloud IoT based Mobile Agent Framework for real-time traffic information acquisition, storage and retrieval, "Smart Devices, Applications, and Protocols for the IoT", Chapter-II, 2019 PP 14-32, IGI Global publisher.

[7] http://juerkkil.iki.fi/files/writings/websocket2012.pdf

[8] http://www.pubnub.com/websockets/

[9] http://www8.garmin.com/aboutGPS/

[10] https://www.xeonbd.com/data-center-bangladesh

[11] https://www.daily-sun.com/printversion/details/427588/Exploring-Bangladesh-Data-Centre

[12] https://cloudscene.com/data-center/bangladesh/dhaka/devotech-dhaka?quote=true

[13] https://www.dhakacolo.com/

[14] M. M. Rahimi, and F. Hakimpour, “Towards A Cloud-based Smart Traffic Management Framework”, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLII-4/W4, 447–453, https://doi.org/10.5194/isprs-archives-XLII-4-W4-447-2017, 2017.

[15] https://www.intel.com.au/content/dam/www/public/us/en/documents/case-studies/big-data-xeon-e5-trustway-case-study.pdf

[16] A. M. d. Souza, C. A. Brennand, R. S. Yokoyama, E. A. Donato, E. R. Madeira, L. A. Villas, “Traffic management systems: A classification, review, challenges, and future perspectives”, Volume: 13 Issue: 4, April 1, 2017, https://doi.org/10.1177/1550147716683612

[17] M. R. Rahman and S. Akhter, “Bi-directional traffic management with multiple data feeds for dynamic route computation and prediction system,” International Journal of Intelligent Computing Research, Special Issue, vol. 7, no. 2, pp. 720-727, 2016.

[18] S. Akhter, M. R. Rahman, and A. Islam, “Neural Network (NN) based route weight computation for bi-directional traffic management system”, International Journal of Applied Evolutionary Computation, vol. 7, no. 4, pp. 45-59, 2016.

[19] S. Nawrin, M. R. Rahman, and S. Akhter, “Exploreing k-means with internal validity indexes for data clustering in traffic management system,” Thomson Reuters Master Journal List and ISI index. International Journal of Advanced Computer Science and Applications, vol. 8, no. 3, pp. 264-272, March 2017.

[20] S. Akhter, S. Sumit, M. R. Rahman “Design and Implementation of an Intelligent Traffic Management System- A Neural Approach”, In book: Exploring Critical Approaches of Evolutionary Computation, Chapter: 4, page 58-79, Publisher: IGI Global, July 2018, 10.4018/978-1-5225-5832-3