A Multi Layer Data Platform Architecture for Smart Cities using oneM2M and IUDX

Shubham Mante*, SVSLN Surya Suhas Vaddhiparthy*, Muppala Ruthwik,
Deepak Gangadharan, Aftab M. Hussain, Anuradha Vattem
Smart City Research Centre, IIIT Hyderabad, India
Email IDs : shubham.mante, suhas.vaddhipar, ruthwik.m, anuradha.vattem@research.iiit.ac.in
deepak.g, aftab.hussain@iiit.ac.in

2022 IEEE 8th World Forum on Internet of Things (WF-IoT) | 978-1-6654-9153-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/WF-IOT54382.2022.10152258

***Abstract*—Smart cities play a vital role in limiting the ill-effects of rapid urbanization on the environment without compromising on benefits such as improving infrastructure, standard of living, and productivity. However, the collection, storage, and sharing of data from the plethora of sensor networks in a typical smart city deployment warrants a well-defined data platform architecture. In this paper, we propose a multi layer architecture compliant with the oneM2M standards and the Indian Urban Data Exchange (IUDX) framework. The proposed architecture consists of Data Monitoring (DML), Data Storage (DSL), and Data Exchange (DEL) layers. The DML employs oneM2M as the middleware platform to achieve interoperability. The DSL uses a multi-tenant architecture with multiple logical databases, enabling efficient and reliable data management. The DEL utilizes standard data schemas and open APIs of IUDX to avoid data silos, and enables secure data sharing. Further, we present a proof-of-concept implementation of our architecture deployed in a university campus using OM2M, PostgreSQL, and Django. Finally, simulations mimicking real-time data insertion and retrieval showed that the DML can handle 600 concurrent users with an average latency under 100 milli seconds. The DSL improved the latency compared to a single database architecture and the DEL could handle 100 concurrent users with zero failed requests.**

***Index Terms*—smartcity; data platform architecture; oneM2M; IUDX; IoT**

I. Introduction

A. Motivation

According to reports [1] from the United Nations Department of Economic and Social Affairs (UN DESA), about 55% of the global population live in urban areas as of 2018, which is projected to reach 68% by 2050. While urbanization of such magnitude vastly improves the Gross Domestic Product (GDP) of any nation and the quality of life of its citizens, it also has the potential to inflict irreversible changes on the environment. Hence, the strategy of urbanization should focus on efficient use of resources and monitoring environmental impact in addition to improving infrastructure, quality of services, and standard of living. The concept of a *Smart City* conforms to these goals, and Internet of Things (IoT) plays a pivotal role in this transformation [2]. Accordingly, metropolitan cities around the world are transforming into smart cities. Parameters such as traffic, weather, and pollution are being monitored in Aarhus (Denmark) [3]. The trajectories of taxis [4] and air quality [5] are being monitored in Beijing (China). Bike sharing data is being monitored in New York and Chicago (USA) [6] while water table level, air quality, etc. are being monitored in Barcelona (Spain) [7]. This data can be used to detect and reduce air pollution, undertake measures to conserve ground water and other natural resources, and reduce traffic congestion. However, sensor networks deployed on a smart city scale produce huge datasets, complicating data accumulation, storage, secure sharability and analytics. Therefore, a well-defined data platform to perform these tasks is a vital requirement in any smart city.

*Shubham Mante and SVSLN Surya Suhas Vaddhiparthy contributed equally to this work.

B. Related works

Multiple architectures targeting smart city applications have been proposed in recent years. Cheng et al. [8] proposed a live City Data and Analytics Platform (CiDAP). The platform employs Aeron, an open source IoT broker, to collect data from multiple sources, a NoSQL database for storage, and a REpresentational State Transfer (REST) based Application Programming Interface (API) to grant access to external applications. Gomes et al. [9] proposed a five layer architecture that utilizes the Pentaho platform for data integration and analytics. The Kettle tool provides the Extraction, Transformation, and Loading (ETL) engine to cleanse the captured data and store it in a uniform format. The subsequent layers include an Apache Cassandra database, an R environment for statistical computing, etc. Liu et al. [10] developed their own ETL tool as part of a data management framework that additionally categorizes data based on sensitivity and adopts different processing (ex: anonymization for private data) and publishing strategies for each level. However, these architectures are designed to efficiently process and store the *heterogeneous* data gathered from various sensor networks, which is also their primary drawback. Eventually, the lack of standard data formats, particularly at the scale of a typical smart city deployment, would lead to data silos.

To resolve this problem, Datta et al. [11] used sensor markup language (SenML) to standardise sensor metadata in a oneM2M based architecture. SenML provides a structured way to encode sensor measurement and additional attributes of a typical IoT device, whereas oneM2M acts as a horizontal service layer that provides common service functions (CSFs) such as data and device management, discovery, security, etc. Additionally, serialized representation of metadata in SenML allows parallel parsing, resulting in efficient oneM2M servers. Jeong et al. [12] leveraged Next Generation Service Interfaces-Linked Data (NGSI-LD) compatible data models and APIs to define a modular Data Platform Architecture (DPA), that mainly aims to provide semantic interoperability. The authors also detail two proof-of-concept implementations including a parking availability prediction system and a COVID support system based on this architecture.

While NGSI-LD, oneM2M, etc. enable semantic and cross-domain interoperability respectively, a smart city oriented DPA should ideally be compatible with a *Data Exchange*[1,2] framework as well. This provides external APIs for seamless and secure data sharing between various stakeholders.

C. Contributions of this work

In this paper, we propose a multi layer DPA that is compliant with the Indian Urban Data Exchange (IUDX)[1] framework and oneM2M standards. The architecture consists of a oneM2M-based Data Monitoring Layer (DML) for seamless data accumulation from various sensor networks of a smart city such as air quality [13], water quality [14], and energy monitoring [15]. This data is stored in the Data Storage Layer (DSL) using a multi-tenant architecture [16] with multiple logical databases, enabling efficient and reliable data management. Finally, the Data Exchange Layer (DEL) enables secure data sharing in a standardised format compliant with IUDX vocabulary. The rest of the paper is organised as follows: Section II describes each layer of the DPA and Section III details our proof-of-concept implementation. We use OM2M [17], an open source service platform compliant with the oneM2M standards in the DML. A PostgreSQL database server is used in the DSL and a Django server acts as the DEL. Further, each layer is deployed on a discrete physical server, enabling independent data transactions (insertion, subscription, retrieval). Finally, the latency and throughput results mimicking real-time data insertion and retrieval are presented in Section IV followed by the conclusion and the future scope in Section V.

II. Proposed Architecture

The proposed architecture consists of a DML, DSL, and a DEL as illustrated in Fig. 1. Multiple IoT nodes post data to the DML at predefined intervals depending on the parameters being monitored. The DML forwards this data using the subscription-notification CSF of oneM2M to the DSL, where it is parsed and ingested into a database. The data can be subsequently accessed by registered clients through the APIs defined in the DEL.

[1]https://iudx.org.in/
[2]https://www.atis.org/scde/

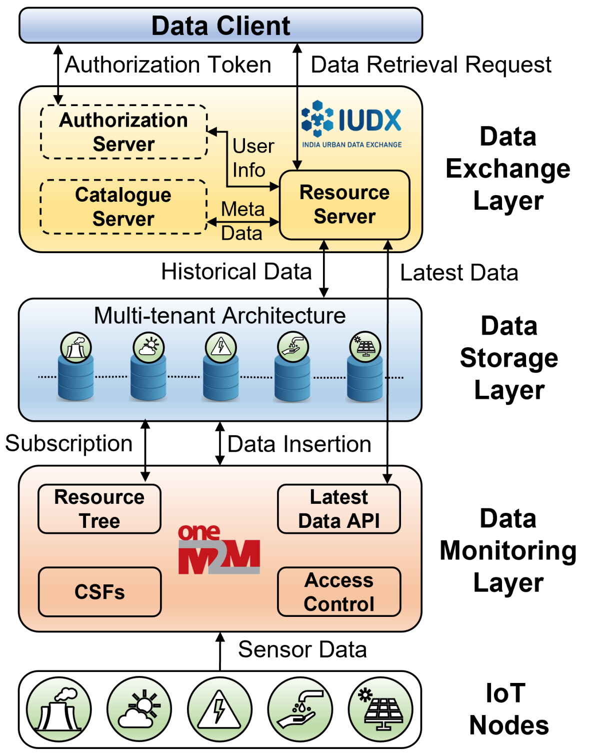


Fig. 1. Illustration of the proposed multi-layered data platform architecture

A. Data Monitoring Layer - DML

The data monitoring layer is based on oneM2M standards. It uses the inbuilt CSFs to accumulate data from various IoT nodes. For instance, every data point generated by a certain water quality node can be stored as a `<contentInstance>` resource inside the respective `<container>` resource. Multiple such containers can act as child resources to an Application Entity (`<AE>`) resource, which corresponds to the entire water quality sensor network. Further, these `<container>` resources can be managed in groups by defining a `<group>` resource. Data forwarding to the DSL is enabled through `<subscription>` resources. Every non-ACP resource is linked to an `<accessControlPolicy>` (ACP) resource via an `<accessControlPolicyID>` (ACPI) attribute. ACPs govern the set of allowed operations that can be performed by a specific user in a given circumstance (time, IP address).

oneM2M also enables discovery and retrieval of resources with options to filter based on type, labels, and content size. Further, each `<container>` resource consists two unique attributes `ol`, and `la` whose values act as Uniform Resource Identifiers (URIs) to retrieve the oldest and the latest `<contentInstance>` respectively. The DML is the first layer to receive any data point, preserving the highest degree of data freshness in the DPA. Hence, every *latestData* retrieval request received by the DEL exploits the `la` attribute provided by oneM2M.

```
{
  "@context": "https://voc.iudx.org.in/",
  "type": [
          "iudx:Resource",
          "iudx:EnvAQM"
        ],
  "id": "research.iiit.ac.in/4786f10afbf48ed5c8c7be9b4d38b33ca16c1d
        9a/iudx-rs-onem2m.iiit.ac.in/iiith-env-aqm/AQ-MG00-00",
  "name": "AQ-MG00-00",
  "label": "Air Quality node at Main Gate",
  "description": "Air quality monitoring device in IIIT-Hyderabad",
  "tags": [
          "environment", "air quality", "air", "aqi", "aqm",
          "pollution", "carbon", "nitrogen dioxide", "co",
          "carbon monoxide", "no2", "pm2.5", "pm10", "nh3",
          "humidity", "temperature"
        ],
  "location": {
              "geometry": {
                          "coordinates": [
                                          78.35142,
                                          17.44599
                                        ],
                          "type": "Point"
                        },
              "type": "Place",
              "address": "IIIT Hyderabad Main Gate"
            },
  "provider": "research.iiit.ac.in/4786f10afbf48ed5c8c7be9b4d38b33
              ca16c1d9a",
  "resourceGroup": "research.iiit.ac.in/4786f10afbf48ed5c8c7be9b4d
                   38b33ca16c1d9a/iudx-rs-onem2m.iiit.ac.in/iiith-
                   env-aqm",
  "itemStatus": "ACTIVE",
}
```

Fig. 2. A sample format of a resource item. This metadata corresponds to an air quality node deployed in IIIT-Hyderabad

B. Data Storage Layer - DSL

The data storage layer uses a multi-tenant architecture with multiple logical databases. This helps in efficient virtualization of available hardware resources across databases. A typical smart city scenario has numerous IoT sensor networks acting as data tenants. Separate logical databases help achieve better data isolation and efficient data management between these sensor networks. Further, they also reduce the impact of data corruption, increase data privacy, and isolate system failures as data transactions such as insertion, deletion, and retrieval are handled separately for each logical database.

C. Data Exchange Layer - DEL

Data exchange layer is based on the IUDX framework for data retrieval using APIs based on NGSI-LD standards. The DEL consists of a Catalogue Server (CS), an Authorization Server (AS), and a Resource Server (RS). The CS and AS are centralised common servers maintained by IUDX and the RS is customised by the data provider.

1) Catalogue Server: The CS maintains the information of the resource groups and resource items. A resource group is a collection of all the resource items associated with it where a resource item refers to an IoT node. A sample format of a resource item is shown in Fig. 2. The CS framework uses a structured vocabulary[3] derived from the Resource Description Framework (RDF) as the base context. The metadata has many user defined attributes such as name, which identifies the unique device name, label and description, which are used for simple description of the device, followed by device static attributes such as location and tags. Finally, the provider attribute identifies the RS paired to the resource item and group.

[3]https://voc.iudx.org.in/

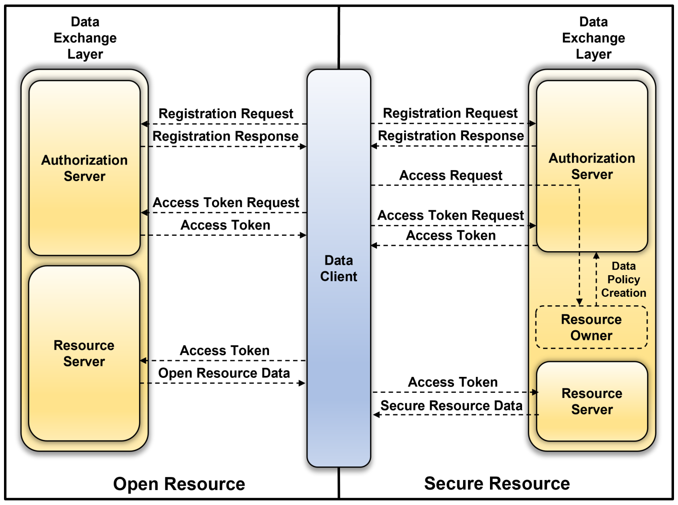


Fig. 3. Data retrieval based on the OAuth 2.0 framework

2) Authorization Server: The AS handles user registration, access policy creation/deletion, and token generation for data retrieval based on the OAuth 2.0 framework. This is detailed in Fig. 3. The data requester must register on the AS to access the IUDX APIs and get the JSON Web Token (JWT) required for data retrieval. To retrieve the data of an open resource, a data consumer can get a token by registering on the AS. However, in case of a secure resource, the registered data consumer can get a JWT only if the data provider has already created the required data policy for the consumer. This allows the data provider to restrict access to secure resources. Fig. 4 illustrates a sample decoded JWT for a secure resource. The data-retriever is given access to APIs, which is denoted by the value of the consent i.e., <cons> key in the decoded JWT.

3) Resource Server: The RS is developed and owned by the data provider. It is primarily responsible for handling data retrieval requests and token verification to prevent unauthorised access. The RS verifies the issuer <iss>, audience <aud>, assign <iat> and expiration <exp> time, resource id <iid>, token requester's role <role> and consent <cons> given to the data consumer in the data access policy in the case of a secure-resource of the received token. The consumer can use the latest API to retrieve the latest data of a resource item, temporal API for historical data, and meta-info API for metadata of a particular IoT node.

III. Implementation

Fig. 5 illustrates the proposed DPA implemented at IIIT-Hyderabad. The DPA handles the monitoring, storage, and retrieval of the data accumulated from various heterogeneous sensor networks and IoT nodes elaborated below.

```
{
   "type": "urn:dx:as:Success",
   "title": "Token authenticated",
   "results": {
                 "sub": "d23a1f88-4073-4cce-be9d-605848e3bfd8",
                 "iss": "authvertx.iudx.io",
                 "aud": "iudx-rs-onem2m.iiit.ac.in",
                 "exp": 1656965348,
                 "iat": 1656922148,
                 "iid": "rg:research.iiit.ac.in/4786f10afbf48ed
                   5c8c7be9b4d38b33ca16c1d9a/iudx-rs-onem
                   2m.iiit.ac.in/iiith-energy-meter",
                 "role": "consumer",
                 "cons": {
                         "apis"
                       }
              }
}
```

Fig. 4. A sample decoded JWT for a secure resource

A. Deployed Sensor Networks

1) Air Quality Monitoring: Several air quality parameters such as particulate matter (pm2.5 and pm10), temperature, relative humidity, and CO concentration are monitored through densely deployed sensor nodes to increase the spatio-temporal resolution of air quality data as detailed in [18].

2) Crowd Monitoring: This sensor network monitors crowding of people to check the number of mask violations to avoid a COVID-outbreak inside the campus.

3) Energy Monitoring: Several energy parameters such as the individual phase currents and voltages, power factor, frequency, energy consumption, apparent and real power are monitored to understand the usage patterns and provide faster resolutions to power outages as detailed in [19].

4) Water Quality Monitoring: Water quality parameters such as total dissolved solids (TDS), and temperature are being monitored to avoid health problems caused by poor quality water, as detailed in [20].

5) Smart Room Monitoring: Occupancy state and energy consumption of a room are monitored to adjust the air conditioning, lighting, and ventilation dynamically.

6) Solar Monitoring: Parameters such as energy generated in a day, signed active power, instantaneous frequency, output power factor, voltage, and current are monitored to efficiently analyse the solar energy generated by solar panels.

7) Weather Monitoring: Multiple weather monitoring stations are deployed to monitor parameters such as solar radiation, temperature, relative humidity, wind direction, wind speed, gust speed and dew point. This data is used for weather forecasting.

B. Data Monitoring Layer

Data monitoring layer uses OM2M, an open-source implementation of the oneM2M standard to collect data from all IoT nodes seamlessly. The horizontal service layer of OM2M enables data interoperability and eases the application development process independent of the underlying communication technologies. The developed resource tree is illustrated in Fig. 6. It consists of an IN-CSE, the root or parent for all the resources in the resource tree. IN stands for Infrastructure Node

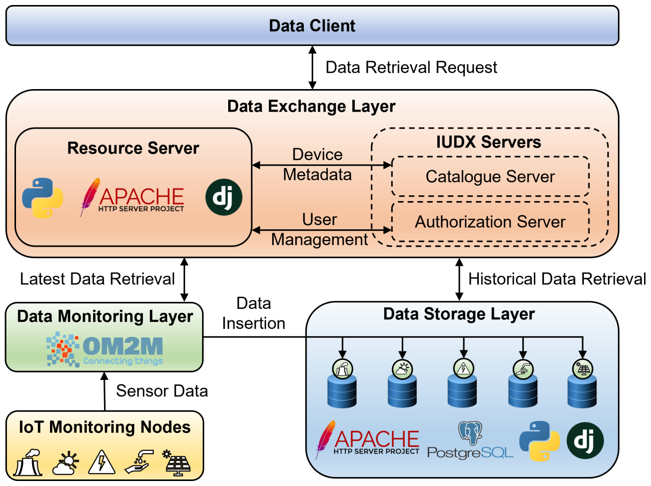


Fig. 5. Implementation of the proposed DPA at IIIT-Hyderabad

and CSE stands for Common Service Entity. The IN-CSE has multiple <AEs> and <cnts>, each <AE> corresponding to a sensor network, and each <cnt> to an IoT node. The <cnt> describes the node's data attributes and stores the incoming live data. Each sensor network has a unique <ACP> to enable controlled data inputs.

For instance, in Fig. 6, AE-CM is the application entity belonging to the Crowd Monitoring (CM) sensor network, which contains six nodes, CM-MG00-00 to CM-VN91-00. These nodes use the *acp-crowd* access control policy for publishing the data. Other sensor networks publish data to the correspondng AEs in a similar manner.

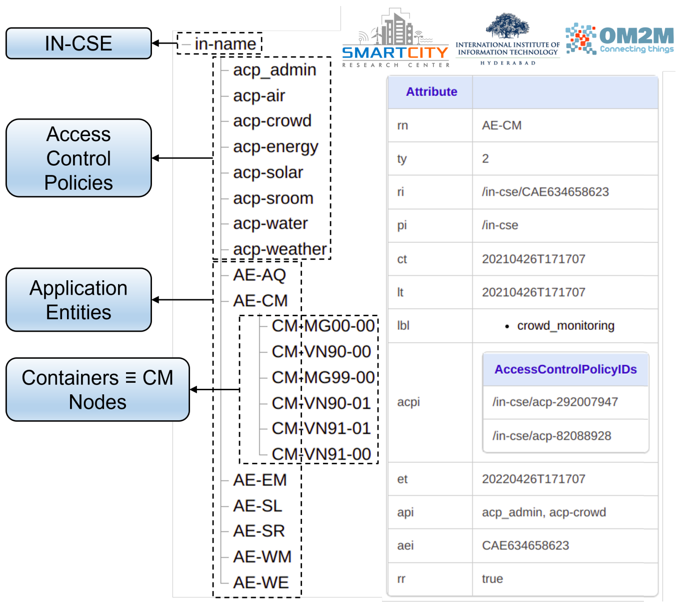
C. Data Storage Layer

Django, an open-source framework for backend web applications based on Python, is used to develop the data insertion APIs in the data storage layer. PostgreSQL, a robust relational database, is integrated with Django to store historical data. The Django API layer subscribes to each IoT node's data container on the OM2M platform using the <subscription> resource. As soon as the OM2M platform receives the data from an IoT node, it forwards the data as a server sent event (SSE) to Django. This data is then parsed to extract the information such as node name, sensor network ID, values of monitored parameters, and timestamp.

As illustrated in Fig. 5, using a multi-tenant multiple logical database architecture, each sensor network has a unique PostgreSQL database instance where the parsed data gets stored.

D. Data Exchange Layer

We developed a resource server based on the IUDX framework in the data exchange layer. The catalogue and authorization servers are standardized central servers for all Indian smart cities maintained by the IUDX organization. The resource server uses Django's REST-API framework to implement five different data retrieval APIs. Fig. 5 illustrates the flow of the data retrieval requests of different APIs as the part of RS implementation. The latest data API request fetches data



Attribute	
rn	AE-CM
ty	2
ri	/in-cse/CAE634658623
pi	/in-cse
ct	20210426T171707
lt	20210426T171707
lbl	• crowd_monitoring
acpi	AccessControlPolicyIDs /in-cse/acp-292007947 /in-cse/acp-82088928
et	20220426T171707
api	acp_admin, acp-crowd
aei	CAE634658623
rr	true

Fig. 6. Illustration of Smart Campus, IIIT-H resource tree, built on the OM2M platform

from the DML through the RS to preserve data freshness. Three temporal data API variants of the RS use the DSL for historical data. Standard temporal API fetches the data based on the timestamps, followed by temporal API with parameter filtering that can filter the requested data for specific parameters. Finally, temporal API with limit-offset acts as a pagination feature for the standard temporal API. The fifth API is the meta-info API which helps the user understand the device information such as sensors, accuracy, and resolution.

Further, in both DSL and DEL, Apache, an open-source, cross-platform web server, is used to host Django along with the mod-WSGI. Apache provides an access control mechanism, web server security, and a high-performance mechanism required for large-scale implementations. The python-based WSGI package acts as a web server gateway that provides an Apache module for communicating and exchanging requests between Django and Apache.

IV. Results and Discussion

Performance analysis has been conducted to evaluate the feasibility, sustainability, and scalability of each layer in the proposed DPA. We have tested the implementation by varying the number of parallel users and evaluating the latency in milliseconds, i.e. the time to process a request, and throughput, i.e. the number of requests served per second, as defined in Equation (1). The experiments were conducted on a computer with an Intel Core-i5-8400 2.80 GHz processor, 8 GB of RAM, and a 64-bit Ubuntu 18.04 operating system.

$$Throughput\ (s^{-1}) = \frac{1000}{Latency\ (ms)} \tag{1}$$

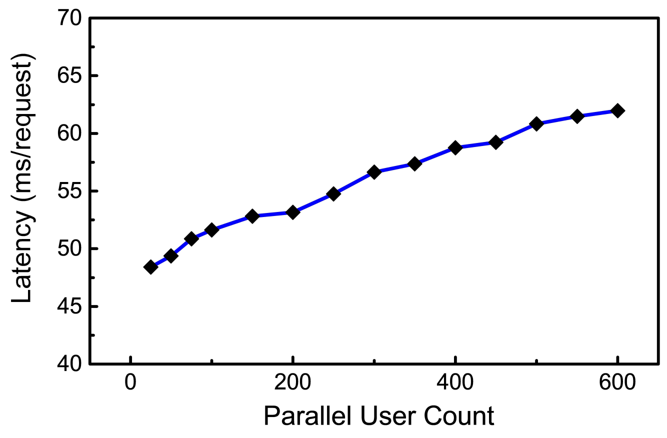


Fig. 7. OM2M retrieval performance characteristic

A. Data Monitoring Layer

The primary purpose of this experiment was to mimic the DML's real-time data insertion and retrieval scenarios. A 12-hour performance test has been conducted on the DML, and the analysis revealed that it could handle 8 parallel users in the data-insertion scenario. As seen in Fig. 7, the retrieval latency increased proportionally with the increasing parallel users, and DML was able to handle upto 600 parallel users with zero downtime in the data-retrieval scenario, while keeping latency well under 100 milliseconds.

TABLE I
Throughput Analysis of Data Exchange Layer

	Type of API				
	Latest	Temporal	Temporal with Parameter Filtering	Temporal with Limit-Offset	Meta-Info
Throughput	3.37	1.91	4.46	4.79	82.17

B. Data Storage Layer

The impetus for this assessment was to highlight the advantages of the proposed DSL architecture for a multi-sensor network scenario. This is reflected by the increased throughput in the standard temporal API scenario. As shown in Fig. 8, the analysis is conducted with two variable parameters: the parallel user count and data points per request. The parallel user count is varied in steps of 25 users, and within each parallel user set, the requested data points are varied from 200 to 600 in steps of 200. The retrieval performance test for all such permutations are conducted on a single database and proposed DSL, where each request effectively retrieves data of three sensor networks simultaneously using the standard temporal API. Clearly, the proposed multi-tenant architecture with multiple logical databases has higher throughput when compared to a single database, as seen in Fig. 8.

C. Data Exchange Layer

Table I summarises the throughput analysis of the different APIs implemented in DEL. The meta-info API outperformed

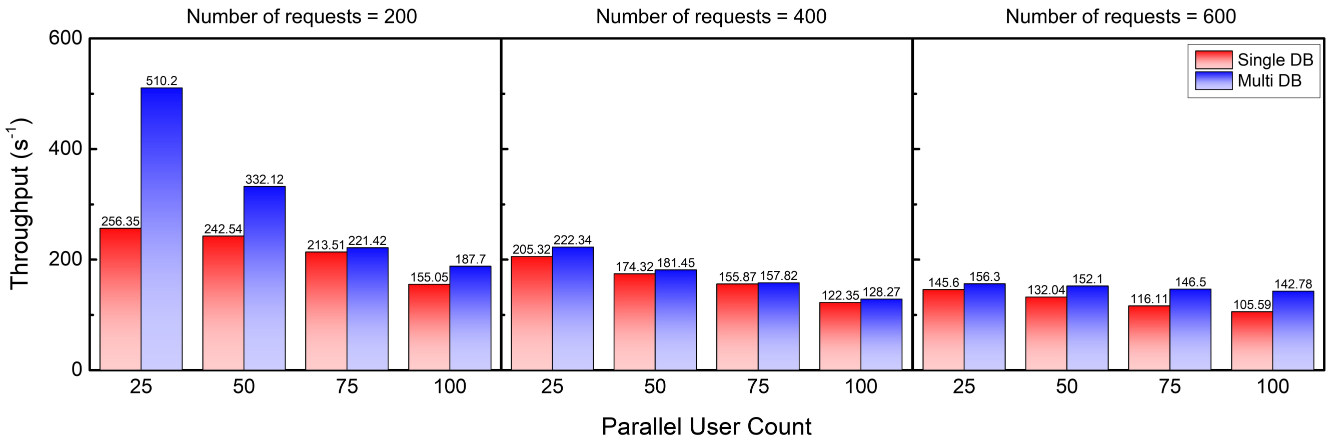


Fig. 8. Throughput analysis of the architecture using single and multiple logical databases.

other APIs because it consists of static device information, followed by the temporal API with filters due to lesser effective data output. The latest data API outperformed the standard temporal API and this can be attributed to the larger data that is retrieved in the temporal API.

V. Conclusion and Future Scope

In this paper, we propose a multi-layer data platform architecture for smart city applications consisting of a Data Monitoring Layer (DML), a Data Storage Layer (DSL), and a Data Exchange Layer (DEL). The DML accumulates data from various sensor networks using a horizontal service layer based on oneM2M standards. The DSL stores this data in a multi-tenant architecture with multiple logical databases to reduce data corruption and isolate system failures. The DEL based on the IUDX framework provides five different APIs to retrieve data. We demonstrate interoperable data management through a proof of concept implementation of the proposed architecture. We evaluated the feasibility and sustainability by analysing latency and throughput of different layers. In the future, a data analysis layer can be integrated into the architecture to gain valuable insights from the collected data. Further, a distributed platform such as Apache Kafka can be integrated to improve the scalability of the architecture.

Acknowledgements

This research was partly supported by the Ministry of Electronics and Information Technology (MEITY) under grant no. 3070665 (2020) as part of the Smart City Living Lab project. We sincerely thank the IUDX team for their support in the development of Data Exchange Layer.

References

[1] UN DESA, “World Urbanization Prospects 2018 - Highlights,” [Online]. Available: https://population.un.org/wup/Publications/Files/WUP2018-Highlights.pdf.

[2] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of Things for smart cities,” *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, Feb 2014.

[3] M. I. Ali, F. Gao, and A. Mileo, “Citybench: A configurable benchmark to evaluate RSP engines using smart city datasets,” in *Proc. Springer 14th Int. Semantic Web Conf.*, Oct 2015, pp. 374–389.

[4] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, “T-finder: A recommender system for finding passengers and vacant taxis,” *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct 2013.

[5] H.-P. Hsieh, S.-D. Lin, and Y. Zheng, “Inferring air quality for station location recommendation based on urban big data,” in *Proc. 21th ACM SIGKDD Int. Conf. on Know. Discovery Data Mining*, Aug 2015, pp. 437–446.

[6] J. Bao, T. He, S. Ruan, Y. Li, and Y. Zheng, “Planning bike lanes based on sharing-bikes’ trajectories,” in *Proc. 23rd ACM SIGKDD Int. Conf. on Know. Discovery Data Mining*, Aug 2017, pp. 1377–1386.

[7] T. Gea, J. Paradells, M. Lamarca, and D. Roldán, “Smart cities as an application of Internet of Things: Experiences and lessons learnt in Barcelona,” in *Proc. 7th Int. Conf. Innov. Mob. Internet Serv. Ubiq. Comput.*, Jul 2013, pp. 552–557.

[8] B. Cheng, S. Longo, F. Cirillo, M. Bauer, and E. Kovacs, “Building a big data platform for smart cities: Experience and lessons from Santander,” in *Proc. IEEE Int. Congr. Big Data*, Jun 2015, pp. 592–599.

[9] E. Gomes *et al.*, “Towards an infrastructure to support big data for a smart city project,” in *Proc. IEEE 25th Int. Conf. Enab. Tech. Infrastr. Collab. Enterpr. (WETICE)*, Jun 2016, pp. 107–112.

[10] X. Liu, A. Heller, and P. S. Nielsen, “CITIESData: A smart city data management framework,” *Knowl. Inf. Syst.*, vol. 53, no. 3, pp. 699–722, Apr 2017.

[11] S. K. Datta and C. Bonnet, “Internet of Things and M2M communications as enablers of smart city initiatives,” in *Proc. 9th Int. Conf. Next Gen. Mobile Appl. Serv. Technol.*, Sep 2015, pp. 393–398.

[12] S. Jeong, S. Kim, and J. Kim, “City data hub: Implementation of standard-based smart city data platform for interoperability,” *Sensors*, vol. 20, no. 23, Dec. 2020.

[13] Z. Hu, Z. Bai, K. Bian, T. Wang, and L. Song, “Real-time fine-grained air quality sensing networks in smart city: Design, implementation, and optimization,” *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7526–7542, Oct 2019.

[14] D. Wu, H. Wang, H. Mohammed, and R. Seidu, “Quality risk analysis for sustainable smart water supply using data perception,” *IEEE Sustain. Comput.*, vol. 5, no. 3, pp. 377–388, Jul 2020.

[15] C. A. Kamienski *et al.*, “Context design and tracking for IoT-based energy management in smart cities,” *IEEE Internet Things J.*, vol. 5, no. 2, pp. 687–695, Apr 2018.

[16] H. Yaish and M. Goyal, “A multi-tenant database architecture design for software applications,” in *Proc. IEEE 16th Int. Conf. Comput. Sci. Eng.*, Dec 2013, pp. 933–940.

[17] M. B. Alaya, Y. Banouar, T. Monteil, C. Chassot, and K. Drira, “OM2M: Extensible ETSI-compliant M2M service platform with self-configuration capability,” *Procedia Comput. Sci.*, vol. 32, pp. 1079–1086, 2014.

[18] C. R. Reddy *et al.*, “Improving spatio-temporal understanding of particulate matter using low-cost IoT sensors,” in *Proc. IEEE 31st Annu. Int. Symp. Pers. Ind. Mob. Rad. Commun. (PIMRC)*, Aug 2020, pp. 1–7.

[19] S. Mante, R. Muppala, D. Niteesh, and A. M. Hussain, “Energy monitoring using LoRaWAN-based smart meters and oneM2M platform,” in *Proc. IEEE Sensors*, Oct 2021, pp. 1–4.

[20] S. U. N. Goparaju, S. S. S. Vaddhiparthy, C. Pradeep, A. Vattem, and D. Gangadharan, “Design of an IoT system for machine learning calibrated TDS measurement in smart campus,” in *Proc. 7th IEEE World Forum Internet Things (WF-IoT)*, June 2021, pp. 877–882.