A Proposal Operational Mechanisms of the IoT-GW Incorporating City OS

Issa Yamanaka[1], Shin-ichi Yamamoto[1,2], Atsuko Yokotani[3], Koichi Ishibashi[4], and Tetsuya Yokotani[5,*]

[1]Graduate School of Electrical Engineering and Electronics, Kanazawa Institute of Technology, Nonoichi Ishikawa, Japan
[2]Tadano Ltd, Takamatsu Kagawa, Japan
[3]Center for Electric, Optic and Energy (EOE) Applications, Kanazawa Institute of Technology, Nonoichi Ishikawa, Japan
[4]Department of Information Science and Engineering, Faculty of Information Science and Engineering, Kanazawa Institute of Technology, Nonoichi Ishikawa, Japan
[5]Department of Electronics and Information Systems Engineering, College of Engineering, Kanazawa Institute of Technology, Nonoichi Ishikawa, Japan
Email: c6401296@st.kanazawa-it.ac.jp (I.S.); shinichi.yamamoto@tadano.com (S.Y.); yokotani@s.ttc.or.jp (A.Y.); k_ishibashi@neptune.kanazawa-it.ac.jp (K.I.); yokotani@neptune.kanazawa-it.ac.jp (T.Y.)
*Corresponding author
Manuscript received May 8, 2025; accepted June 13, 2025; published June 20, 2025

***Abstract*—Recently, various Internet of Things (IoT) services have been considered for implementation for facilitating smart cities. For instance, a horizontal approach has been explored to reduce equipment and operational costs. A City Operating System (OS) is one scheme for achieving this approach. The advantages of a City OS include the independence between connected end systems and services, and the ability to share collected data among multiple services. Currently, some City OS are generally implemented in a centralized configuration (e.g., cloud based configuration). However, there are also services that require data only in closed areas with low latency. In this study, an architecture that enables the coexistence of closed base data and wide-area base data, according to the type of service is proposed. Specifically, authors propose a gateway that incorporates with a City OS and also an operational mechanism that includes a function for sorting data from end systems to both local and global platforms. This proposal contributes to facilitating the realization of smart cities in which various services are layered.**

***Keywords*— City OS, Gateway, IoT, Smart city**

I. Introduction

City OSs are currently attracting attention for solving the challenges of smart city implementation [1, 2]. City OS provides an abstraction of smart city devices. This allows services in various fields to handle data from smart city devices [3].

In a conventional city OS, data integration is primarily implemented in the cloud to achieve efficiency, cost reduction, and scalability [4]. However, there are disadvantages to using the cloud in a city OS. The response performance is poor for services that require immediate feedback processing based on received data (e.g., services with actuation) [5]. To overcome this disadvantage, we consider that it is better to process data in the IoT-GW if the service is locally closer from the viewpoint of communication congestion to the cloud. Therefore, the authors propose an IoT-GW equipped with a city Operating System (OS) that debloats and operates within the local network. This gateway, equipped with the city OS, abstracts devices and enables feedback in the local area without passing services that require immediate response to the cloud.

In this paper, Section II provides an overview of the city OS, and Section III describes the evolution of gateway requirements. The functional structure of the IoT-GW is described in Section IV and its operational flow is described in Section V.

II. Overview of City OS

Traditionally, service provision has been vertically integrated, relying on individual devices and data formats in each field and region. However, in city OS, the first step is to abstract devices, thereby eliminating the dependency between services and devices. This enables specific services to handle data across diverse areas without being tied to specific devices. Furthermore, by unifying data formats and interfaces, the mutual use of data between cities becomes possible, and efficient provision of services can be realized. Specifically, the following three points can be realized [6].

(1) Service portability: Services deployed in one region may be readily redeployed in another. In other words, service providers no longer need to be bound to a region.

(2) Service diversification: This makes it easy to add or remove services in a given region.

(3) Advancement of services: Through the mutual use of data, it may be easy to refer to data from other regions, and the services themselves can be expected to be advanced.

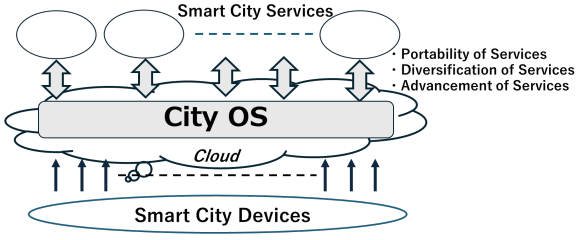

Fig. 1. System configuration with city OS.

The system configuration incorporated in to a city OS is shown in Fig. 1, where a conventional city OS is incorporated in the cloud. The data extracted from the smart city devices in each area are managed by the city OS in the cloud to realize a variety of smart city services. The internet is widely used to connect smart city devices to the cloud, however not all end devices are directly connected to the internet. Therefore, Gateways (GWs) are installed in each area to aggregate local communications with end devices and multiplex data as needed, and perform relay functions to forward data to the cloud.

FIWARE is often used as one of the city OSs, which is an infrastructure software developed and implemented under the next generation internet Public-private Partnership Program (FI-PPP) of the European Union (EU) to link various types of data [7, 8]. It is also a collection of software modules called the Generic Enabler (GE), and Orion Context Broker (OCB) is a GE which characterize FIWARE [1]. The FIWARE architecture and its key components are illustrated in Fig. 2. Other representative components include Cygnus, which stores historical data registered in OCB in a Data Base (DB), and Wirecloud, which receives and visualizes data from OCB [9]. These data exchanges were performed using the open standard FIWARE Next Generation Service Interface (NGSI) API [8]. Another IoT Agent automatically converts data from smart city devices into an NGSI data model [9].

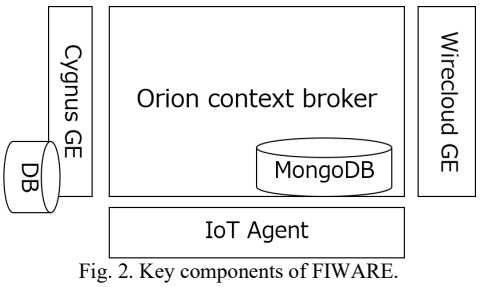

Fig. 2. Key components of FIWARE.

III. Transition of Gateway Requirements

This approach was adopted because Home Gateway (HGW) provide a cost-effective and practical foundation for large-scale deployment, particularly in environments such as households and small facilities, where IoT services are frequently implemented. Although server-based gateways with more advanced capabilities exist, HGW-based architectures are considered more suitable in terms of deployment density and service pricing. Therefore, the proposed architecture builds upon the HGW technology to ensure both scalability and affordability. Fig. 3 illustrates the gateway evolution is illustrated based on HGW technology.

Each phase is described as follows. In Phase 1, communication was limited to the inside of a building, as the focus was on home device control and security applications, and low-speed, pay-as-you-go internet connections via Integrated Services Digital Network (ISDN) were the norm. In Phase 2, the spread of Asymmetric Digital Subscriber Line (ADSL) enabled flat-rate broadband connections, and the gateway played the role of a broadband router connecting multiple devices in the home to the internet. In Phase 3, Fiber to the Home (FTTH) significantly improved the communication quality, and as Voice over Internet Protocol (VoIP) became increasingly popular, gateways were required to support real-time voice communication and advanced packet processing, such as the priority processing of voice data. In Phase 4, the advent of Next-Generation Networks (NGNs) enabled the "triple play" [10] of telephony, video distribution, and Internet access, and support for all these services became necessary. In addition, support for home ICT services [11] such as IPv6 support and energy management, is required, and a flexible service delivery platform with Open Services Gateway Initiative (OSGi) was introduced. In Phase 5, the gateway, as the center of the IoT services, was equipped with a physical interface that enabled connection to various IoT devices and the ability to link application data with servers [12]. In Phase 5, the gateway converts the data into a FIWARE-compliant data model and processes the data in the cloud, making data from a variety of fields available for a variety of services. However, the gateway in Phase 5 was unable to provide the same level of service as the urban gateway.

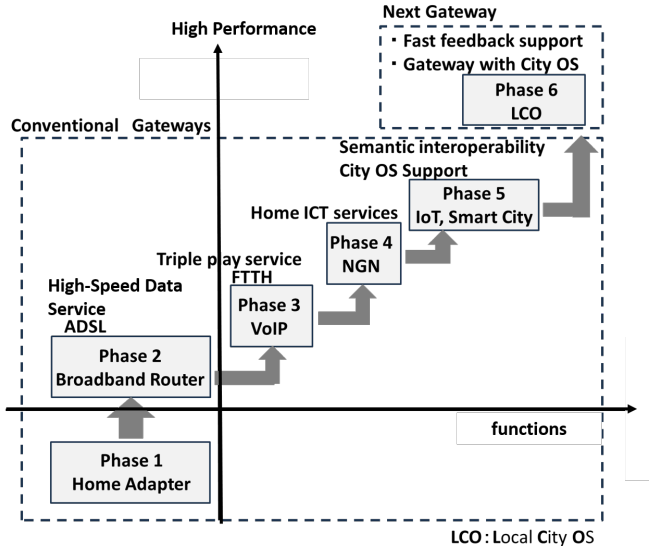

Fig. 3. Transition of gateway requirements based on [12].

Although the Phase 5 gateway supports City OS, it is implemented in the cloud and is not suitable for high-speed feedback such as actuator-supported services. The authors believe that a new phase of 6 GW with high-speed feedback is required. In Phase 6, the GW must be able to support actuator-supported services, reuse data used in actuator-supported services to prevent congestion and cost increases for devices such as sensors and cameras, and be customizable to flexibly support various services. Therefore, the authors propose a feature of the city OS, data abstraction, that enables the reuse of data across different services. Furthermore, the authors believe that by incorporating the open-source software, city OS, into the gateway, these requirements can be met simultaneously.

In this study, the authors propose a new gateway for Phase 6, in which the gateway is equipped with a "Local City OS (LCO)," which is a function to equip a city OS in the gateway, in addition to the functions of the conventional gateways up to Phase 5.

IV. Result and Discussion

The proposed IoT Gateway (IoT-GW) requires two key mechanisms: one to enable data from Smart City Devices (SCDs) to be processed in the cloud, and another to enable local data processing. The first mechanism is achieved using the existing Phase 5 functions, including temporary data aggregation, data model conversion and aggregation, data model transmission, and the gateway function. The second mechanism is realized by incorporating a message broker into gateway, which is a core function of the city OS. To support both mechanisms within a single gateway, an Interface (IF) equipped with a data analysis function is introduced, because data sorting depends on prior analysis. An overview of these functions is shown in Fig. 4. Six functions of Local City OS (LCO) (see Figs. 4a-f) and one

function of Interface (IF) (see Fig. 4(g)) (see Fig. 4(B)) of the IoT-GW are described below: LCO is in addition to conventional gateway functions until Phase 5, which is a function for installing the City OS in the gateway.

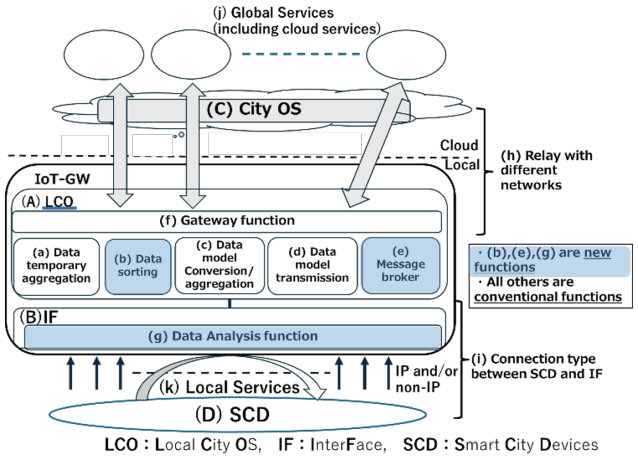

Fig. 4. Proposed IoT-GW architecture.

The new functions of the gateway in Phase 6 of Section III are shown in Fig. 4(b), (e), and (g), whereas the other functions are the same as those in Phase 5 of the HGW.

First, functions (a)–(f) of the LCO are described below:

(a) Temporary data aggregation [conventional]

This function temporarily aggregates data generated in the local area of the LCO without sending them to the cloud. This enables the provision of services close to the gateways.

(b) Data Sorting [new function]

Following the data analysis by IF, the data are classified into those used for local services (services terminated within the gateway) and those used for global services.

(c) Data model conversion and aggregation [conventional]

The various forms of data sent to the LCO are abstracted and converted into a data model that is compatible with the City OS. Furthermore, multiple abstracted datasets can be aggregated into a single data set model.

(d) Data Model Transfer [conventional]

The data model generated in (c) is transferred to a message broker or City OS in the cloud based on the sorting results in (b).

(e) Message broker function [new function]

This is a broker function (City OS function) for data messages sent to the LCO.

(f) Gateway function [conventional]

Fig. 4(h) shows the relay between the IoT-GW and different networks such as mobile communications.

The role of IF (see Fig. 4(B)) is as follows: IF is also a new function in Phase 6.

(g) Data analysis function [new function]

Data from the Smart City Devices (SCD) are analyzed by the Interface (IF) (see Fig. 4(B)) in the gateway and the analysis results were added to the data from the SCD. Details are given in the next section; the connection between the SCD and the IF (see Fig. 4(i)) is assumed to be IP or non-IP communication (e.g., I2C communication [13]), and these processes are performed in the IF (see Fig. 4(B)).

V. Proposed Operational Mechanisms

This section describes the functional operation flow of the IoT-GW proposed in this study. First, the operational flow of the entire function is described, followed by that of the newly added data sorting function.

A. Overall Operational Flow of IoT-GW

The overall operational flow of the IoT-GW is shown in Fig. 5. The overall operational flow of the IoT-GW proposed by the authors is shown step-by-step, i.e., (i)-(vii) in Fig. 5.

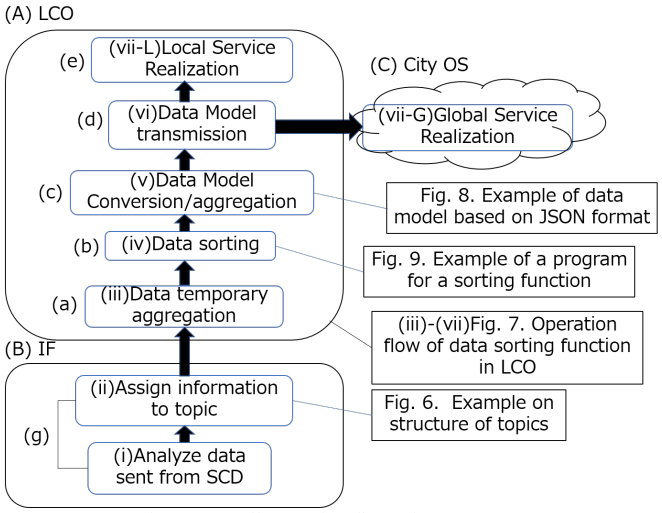

Fig. 5. Overall operation flow of IoT-GW.

(i) At each interface (IF, see Fig. 4(b)), data transmitted from the SCD (see Fig. 4(d)) were analyzed as useful information using a pre-built algorithm. Data analysis methods included comparison with threshold values and machine learning. Data analysis was also used to determine urgency and locality (Fig. 4(g)).

(ii) Information such as urgency and locality are assigned to "topics." A "topic" refers to the name attached to the data. Information, other than urgency and locality, may also be included in the topic. For instance, the sensor location information is appended to a topic. Fig. 6 shows an example of attaching information to a topic on an interface (IF) (see Fig. 4(g)).

interface/(a) sensor name/ (b) location/ (c) urgency/ (d) locality
Example :
interface/Sensor1/location1/emergency_yes/LocalData_yes

Fig. 6. Example on structure of topics.

(iii) The SCD data are temporarily aggregated to a topic in the LCO (see Fig. 4(a)).

(iv) Based on the topic information, the data to be sent to the message broker in the LCO or city OS in the cloud are sorted (see Fig. 4(c) and 4(b)).

(v) Based on the attributes of the sorted data (e.g., urgency and locality), the data are abstracted and a common data model is created that is easy to handle for each service.

(vi) The data model is sent to the message broker in the LCO or the City OS in the cloud (Fig. 4(d)).

(vii) Realize local services (Fig. 4(k) and 5(vii-L)) and global services (Fig. 4(j) and 5(vii-G)) using data stored in the message broker in the LCO (Fig. 4(e)) and in the City OS in the cloud (Fig. 4(c)).

B. Operation Flow of Data Sorting in LCO

Among the new features added in Phase 6, data sorting

(Fig. 4(b)) is described in detail. Fig. 7 shows the data sorting in the LCO. The operational flow of the data sorting function shown in Fig. 7(1)–(5) is described as follows.

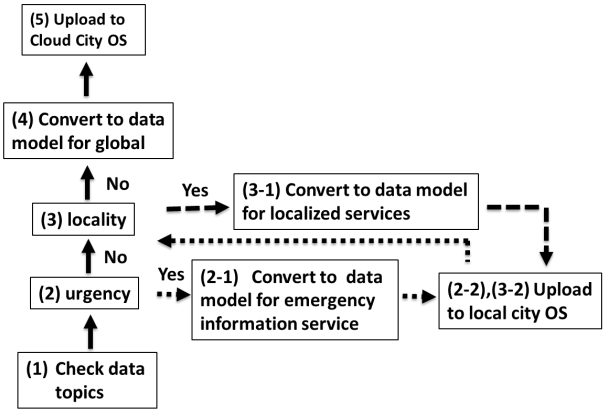

Fig. 7. Operation flow of data sorting function in LCO.

(1) Sort the data by selecting data from the temporary tally (Fig. 6(a)) and checking the attributes of the data topic.

(2) Check the urgency; if yes, convert the data to a data model for emergency information service (2-1) and send it to the LCO (2-2). If no, go to (3).

(3) Checks for locality; if yes, converts the data to a data model for localized services (3-1) and sends the data model to the LCO (3-2); if no, go directly to (4).

(4) Convert the data used for global services into the data model for global services.

(5) Upload the data model for global services to City OS in the cloud.


```
{
  "id": "urn:ngsi-ld:ObjectDetectionSensor:001",
  "type": "ObjectDetectionSensor",
  "dataissued":{
    "type": "Text",
    "value": "2024-07-12 16:02:56.343000"
  },
  "location":{
    "type": "geo:json",
    "value":{
      "type": "Point",
      "coordinates":[132.5304138,34.2567817]
    }
  },
  "CurrentDetectionStatus":{
    "type": "boolean",
    "value": "true"
  },
  "DetectionType":{
    "type": "Text",
    "value": "person"
  },
}
```


Fig. 8. Example of data model based on JSON format.

The data model is often described in JavaScript Object Notation (JSON) format using an international standard for Application Programming Interfaces (APIs) called Next Generation Service Interface (NGSI) [14]. Multiple data points are aggregated into a single data model (Fig. 4(c)). An example data model is shown in Fig. 8.

The authors considered using a Python Program to perform data sorting for the IoT-GW implementation, as shown in Fig. 9. Fig. 9 shows an example of such a program. The program reads the sent topic data and divides them into hierarchies. The hierarchy determines the keywords, checks the keywords against the database, and decides whether to send the data to the LCO or to City OS in the cloud.

This data sorting avoids sending data that is only needed for local services to the city OS in the cloud, preventing network congestion.

```
t_p_c = tpc.split("/")
keyword= t_p_c[1]
dict_from_csv = {}
with open('db/db.csv', mode='r') as inp:
  reader = csv.reader(inp)
  dict_from_csv = {rows[0]:rows[1] for rows in reader}
local_global_flg = int(dict_from_csv[keyword])

if local_global_flg == 0:
  fiware_IP = fiware_IP_local
elif local_global_flg == 1:
  fiware_IP = fiware_IP_global
else:
  fiware_IP = fiware_IP_local
  fiware_IP2 = fiware_IP_global
  flg = 1
```

Fig. 9. Example of a program for a sorting function.

VI. Conclusion

In this paper, the authors have proposed operational mechanisms for an IoT-GW with a City OS that provides fast feedback, along with its functional configuration. The proposed method enabled device abstraction and rapid feedback within a local area. This facilitate the sharing of devices and implementation of services using actuators. In addition, by avoiding reliance on the cloud for data processing, communication path congestion is reduced.

A Proof of Concept (PoC) is planned to verify the technical feasibility of the proposed approach. In addition, verification from the user perspective by applying actual services has been performed to assess its practical effectiveness.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Issa Yamanaka initiated the research and drafted the initial manuscript. Shin-ichi Yamamoto refined and deepened the conceptual framework. Atsuko Yokotani reviewed the manuscript and contributed to the overall enhancement of its content. Koichi Ishibashi checked and revised the language and expressions. Tetsuya Yokotani, as the supervising professor, provided skill-based guidance and conducted a comprehensive review of the work. All authors had approved the final version.

References

[1] N. Fujita, K. Fujita, and O. Tashiro, "Urban OS supporting the spread and development of smart city," *Smart Construction Research*, vol 1, no. 1, pp. 11–21, 2023.

[2] E. A. Nuaimi, H. A. Neyadi, N. Mohamed, and J. AI-Jaroodi., "Applications of big data to smart cities," *Journal of Internet Services and Applications*, vol. 6, 25, Dec. 2015.

[3] I. A. T. Hashem, V. Chang, N. B. Anuar, K. Adewole, I. Yaqoob, A. Gani, E. Ahmed, and H. Chiroma, "The role of big data in smart city,"

International Journal of Information Management, vol. 36, no. 5, pp. 748–758, 2016.
[4] A. K. Sandhu, "Big data with cloud computing: Discussions and challenges," *Big Data Mining and Analytics*, vol. 5, no. 1, pp. 32–40, 2022.
[5] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5031–5044, 2019.
[6] A. Detti, H. Nakazato, J. A. M. Navarro, G. Tropea, L. Petrucci, J. A. S. Segado, and K. Kanai, "VirIoT: A cloud of things that offers IoT infrastructures as a service," *Sensors*, vol. 21, no. 19, 6546, 2021.
[7] M. Bauer, "FIWARE: Standard-based open source components for cross-domain IoT platforms," in *Proc. IEEE 8th World Forum on Internet of Things (WF-IoT 2022)*, 2022, pp. 1–6.
[8] F. Cirillo, G. Solmaz, E. L. Berz, M. Bauer, B Cheng, and E. Kovacs, "A standard-based open source IoT platform: FIWARE," *IEEE Internet of Things Magazine*, vol. 2, no. 3, pp.12–18. 2019.
[9] FIWARE catalogue. [Online]. Available: https://www.fiware.org/catalogue/#componets
[10] S. Karapantazis and F. N. Pavlidou, "VoIP: A comprehensive survey on a promising technology," *Computer Networks*, vol. 53, no. 12, pp. 2050–2090, 2009.
[11] T. Ishihara, K. Sukegawa, and H. Shimada, "Home gateway enabling evolution of network services," *FUJITSU Sci. Tech. J.*, vol. 42, no. 4, pp. 446–453, 2006.
[12] E. Yoshida, T. Yokotani, and K. Ishibashi, "Concept and development of next phase gateway applied to internet of things," in *Proc. 2021 11th International Workshop on Computer Science and Engineering (WCSE 2021)*, 2021, pp. 13–21.
[13] J. Mankar, C. Darode, K. Trivedi, M. Kanoje, and P. Sharare, "Review of I2C protocol," *International Journal of Research in Advent Technology*, vol. 2, no. 1, pp. 474–479, 2014.
[14] M. Bauer. E. Kovacs, A. Schulke, N. Ito, C. Criminisi, and L.W. Goix, "The context API in the OMA next generation service interface," in *Proc. 2010 14th International Conf. on Intelligence in Next Generation Networks*, 2010, pp. 1–5.