Knowledge Graph based Internet of Things Middleware

Knowledge Graph based Internet of Things Middleware

Cheng Xie, Di Liu, Yun Yang*, Po Yang‡, Beibei Yu*, Zhibo Chen†, Qiang Feng§, Jiqin Peng§
*School of Software, Yunnan University, Kunming, China
†Law School, Yunnan University, Kunming, China
‡Department of Computer Science, Liverpool John Moores University, Liverpool, UK
§Zhongmao Environment Energy Technology Investment, Kunming, China

***Abstract*—Internet of Things (IoT) provides ubiquitous intelligence and pervasive interconnections to diverse physical objects. A key technology to seamlessly integrate different IoT devices into an IoT system is IoT middle-ware, a software system layer designed to be the intermediary between IoT devices and applications. However, two issues, *communication gap* and *heterogeneous access*, prevent the existing IoT middle-ware from effective application in an IoT system with heterogeneous standards and interfaces. To address this problem, inspired by graph based knowledge system for eliminating heterogeneity in business systems, we in this paper propose a knowledge graph based Multi-Layer IoT middle-ware. The proposed multi-lay IoT middle-ware introduce a new lay to bridge the gap between IoT devices with different communication protocols and is able to uniformly manage all IoT devices by using an IoT knowledge graph. We evaluate the applicability of the proposed approach by a real-life IoT project, a remote monitoring project of rural sewage treatment stations located at Yunnan Province, China. We find that the proposed approach effectively resolves the communication gap and heterogeneous access problems occurred in the system.**

***Index Terms*—Internet of Thing, Knowledge Graph, Ontology, Web Service, IoT Application, IoT Middleware**

I. Introduction

The Internet of Things (IoTs) is emerging as a new information revolution in the new Internet era Numerous devices connected to Internet significantly influence different aspects of our life and work Sophisticated IoT systems implement fruitful functionalities by combining and integrating different IoT devices into a large-scale system.

However, IoT devices with different functions are usually manufactured by different companies, so they may use completely different access standards, thereby demonstrating prominent heterogeneity in terms of access method. Therefore, how IoT devices with different access methods effectively communicate with each other in a complicated system has become one of the main challenges faced by IoT applications [1]. To handle this challenge, previous research proposed the IoT middle-ware to bridge the gap between heterogeneous IoT devices and other systems or applications . Although the IoT middleware methodology shows its feasibility in theory, it still encounters several problems in practical application.

- **The communication gap**: some IoT devices are small in size and have limited computing and communication capabilities (e.g., temperature sensors, humidity sensors, etc.). They only support *field communication protocols* such as RS485, Modbus, etc, and are unable to support *IoT communication protocols* such as CoAP, MQTT, ZigBee, etc. Thus, those IoT devices can not directly access IoT middleware.
- **The heterogeneous access:** the Machine-to-Machine (M2M) communication plays a key role in the IoT middleware framework. However, there are many different M2M standards (oneM2M, LWM2M, FIWARE, etc.) and they are not all compatible with each other, and different IoT devices may conforms to different M2M standards, thereby leading to the heterogeneous access problem. Even in some cases, some devices in the system use a customized standard instead of any existing communication standard, that further exacerbates the heterogeneous access problem.

Therefore, in order to successfully apply the IoT middleware approach in an IoT system with complex communication networks, we need to find an effective solution to the communication gap problem and heterogeneous access problem.

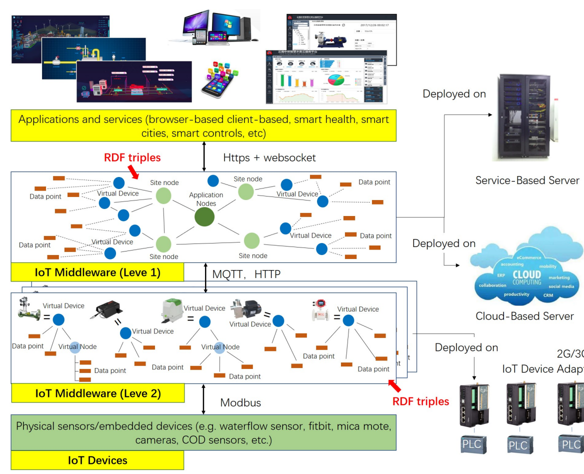

Fig. 1. The knowledge graph based IoT middleware for industry application

In this paper, we propose a novel multi-layer IoT middleware method based on knowledge graph for IoT devices management in a complex IoT system. The illustration of the new method is shown in Fig.1. The major contributions of this new method are twofold:

1) Similar to GraphQL, we propose a novel method based on knowledge graph theory to uniformly manage the communication of all devices. The proposed method maps an IoT device to a sub-graph in a knowledge graph. As a result, the operation of the device no longer calls the APIs of the IoT devices, but to query or modify the corresponding edges and nodes in the mapped graph. Then, the communication and interoperation between different IoT devices can be done via the attribute manipulation in a knowledge graph. Therefore, the heterogeneous access problem mentioned above is effectively solved;

978-1-7281-2927-3/19/$31.00 © 2019 IEEE

2) We introduce a new layer namely Level-2 (L2 for short) IoT middleware. The new layer serves as the bridge between the IoT devices which only support field communication protocols (e.g., RS485 or ModBus) and the existing IoT middleware which deploy some IoT protocols (e.g., MQTT, HTTP and other IoT protocols). L2 IoT middleware lay effectively mitigate the communication gap issue.

We evaluate the applicability of the proposed multi-layer IoT middleware by a real-life IoT project, a remote monitoring project of rural sewage treatment stations located at Yunnan Province, China, more details in Section IV. We find that the proposed approach effectively resolves the communication gap and heterogeneous access problems occurred in the system. At present, we have successfully implemented the proposed approach in 15 sewage treatment stations around the province. This success paves the way for the future application of our approach in different IoT systems with the same problem.

II. Related Works

A. IoT Middleware

While IoT offers numerous exciting potentials and opportunities, it still remains challenging to effectively manage 'things' to achieve a seamless integration of the physical world and the cyber ones [2]–[4]. An enabling technology in integrating IoT devices in a system is IoT middleware, which is a software system layer designed to be the intermediary between IoT devices and applications [1]. The state-of-the-art IoT middlewares, such as Hydra [5], GSN [6], Google Fit and Xively, are all based on Web Service (or API based). These middlewares apply service composition and discovery to handle heterogeneity of IoT devices which requires a lots of manual and programming efforts. Moreover, these middlewares use WAN communication protocols (HTTP, SOAP, MQTT, ZigBee, CoAP, etc.) that can not be applied on many small IoT devices which only support field communication protocols (RS485, ModBus/RTU, etc.), i.e., the communication gap issue mentioned in Section I. In the paper, in comparison with the existing IoT middleware frameworks, we propose a multi-layers IoT middleware based on knowledge graph to deal with the communication gap and heterogeneous access issues.

B. Knowledge Graph

Knowledge graph is widely used in question answering [7], [8], natural language processing [9], affective analysis [10], association prediction [11] and other artificial intelligence applications, and has achieved outstanding results. In recent years, many companies have begun to use knowledge graph based system for their business related systems as well.

1) Structure of a knowledge graph: Popular knowledge graphs such as DBpedia [12], YAGO [13], NELL [14] and KnowledgeVault [15] are composed of a base ontology, multiple domain-specific ontologies and instances. The base ontology of a knowledge graph is used to uniform the "terms" and "relations" in "top-level". The domain-specific ontology of a knowledge graph is used to specify the "terms", "relations" and restrictions in a certain domain. The instances are the data of the knowledge graph.

III. The Method

A. IoT Knowledge Graph

According to the basic definition of Knowledge Graph, IoT Knowledge Grap consists of three parts: IoT base ontology, IoT domain ontology and IoT instance.

1) IoT base ontology: IoT base ontology can be understood as a specification or standard to unify the naming of relationships and attributes of IoT devices. Based on oneM2M base ontology (Y.4500.12(18)_F01) [1], we define the basic structure of an IoT device in the Knowledge Graph from a high level, as shown in Fig. 2.

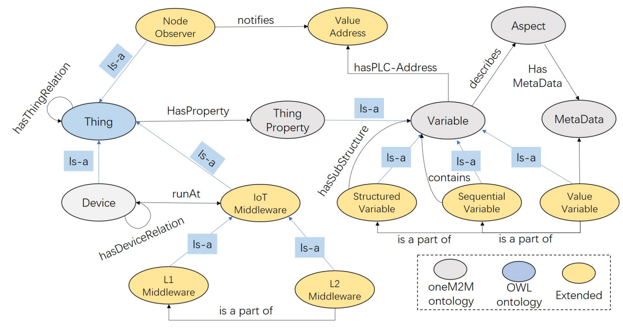

Fig. 2. The base ontology of the top level knowledge graph of the framework

2) IoT domain ontology: IoT base ontology defines the basic structure and relationship of IoT devices in an abstract domain. In contrast, IoT domain ontology is designed to unify attributes, names, data points and interface of IoT devices in a specific domain. In this work, oneM2M domain ontology[2] is applied with IoT base ontology. The oneM2M domain ontology intends to provide the unified means in the oneM2M system by defining an information model for the specific domain devices such as thermometer, flow-meter, water sensor, smoke sensor, refrigerator, air conditioner and so on. Fig. 3 gives an example of oneM2M domain ontology for describing a thermometer.

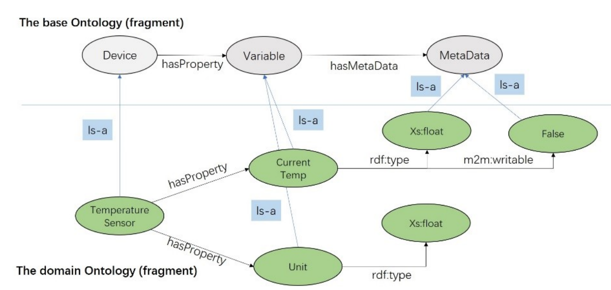

Fig. 3. An example of oneM2M domain ontology for describing a thermometer

3) Device URI generation: IoT device mapping to IoT instance graph requires assigning a globally unique identifier to the device. The common practice is to use the device's URI as the unique identifier. The standard definition of URI is as follows.

$$URI = [ApplicationDomainName]/[BaseOntology] \\ /[DomainOntology]/[DeviceID]$$

First, a device is mapped to a Device class in IoT base ontology. Then, the name of the device is mapped to the standard name in IoT domain ontology, which alone with the Device class of the base ontology serves as the localpath

[1]http://www.onem2m.org/technical/published-drafts
[2]http://www.onem2m.org/component/rsfiles

of the URI. The serial number of the device is placed at the end of the URI as a localname. We can access the URI directly to obtain the root node of the device in IoT knowledge graph. For instance, querying <https://www.zm-iot-platform.com/Device/thermometer/300218090078> will be able to gain the root node of the thermometer in IoT knowledge graph.

4) Value Variable: Value Variable refers specifically to the primitive data types defined in W3C XSD 1.1, such as integers, floating point numbers, time, characters, etc. The mapping rules for this type are as follows:

$$\begin{aligned} URI =& [ApplicationDomainName]/ValueVariable \\ & /[DomainOntology]/[Value] \end{aligned}$$

According to the mapping rules, the attributes of the device are first mapped to a ValueVariable in BaseOntology. Subsequently, the name of the attribute is mapped to the standard attribute name in DomainOntology, which together with ValueVariable acts as the localpath of the URI. The value of the attribute is placed at the end of the URI as a localname. For example, a value 23° collected by a thermometer is mapped to a node of IoT knowledge graph as <https://www.zm-iot-platform.com/ValueVariable/CurrentTemp/23° >.

5) Structured Variable: Structured Variable is a combination of one or more sets of data that cannot be mapped directly to primitive data type. Structured Variable is actually a tree structure with non-leaf nodes representing a sub-structure and leaf nodes storing specific data. In IoT base ontology, Structured Variable is defined as a subclass of Variable. Structured Variable uses "hasSubStructure" to link with substructures. In the knowledge graph, it is expressed as: <StructuredVariable> <hasSubStructure> <Variable>. For leaf nodes, we can map directly using ValueVariable mapping rules. For non-leaf nodes, we define the following URI mapping rules.

$$\begin{aligned} URI =& [ApplicationDomainName]/StructuredVariable \\ & /[DomainOntology]/[AutoID] \end{aligned}$$

6) Sequential Variable: Sequential Variable is a set of Variables. It can be ordered or disordered. Sequential Variable cannot be mapped directly to Value Variable. In IoT base ontology, Sequential Variable is defined as a subclass of Variable that contains multiple Variables. These Variables can be ValueVariable, Structured Variable, or Sequential Variable. Sequential Variable uses "contains" relationship to link with the Variable it contains, and obtains the next Variable orderly through "hasNext" relationship. The URI mapping rules between Sequential Variable and knowledge graph are as follows:

$$\begin{aligned} URI =& [ApplicationDomainName]/SequentialVariable \\ & /[DomainOntology]/[AutoID] \end{aligned}$$

IV. The Case Study

A. The project description

We evaluate the applicability of the proposed multi-layer IoT middle-ware by implementing it in a real-life IoT project. The main object of this project is to provide an automatic sewage treatment systems for some rural areas in Yunnan Provice, China. All sensors and controllers are required to connect to an IoT environment, such that an administration center is able to monitor the quality of water, to make a timely decision in an emergent situation and to control all actuators (devices) remotely.

This sewage treatment system has the *communication gap* and *heterogeneous access* issues mentioned in Section I. The sewage treatment system consists of two groups of blowers, a group of hoist pumps, a group of self-priming pumps, a group of backwashing pumps, two groups of back-flow pumps, two solenoid valves, two COD sensors, a flow sensor, a PH sensor, a temperature sensor, and a turbidity sensor. However, these pumps and sensors do not have enough computing and communication capabilities, and cannot be directly connected with the existing IoT middle-ware. At the same time, some devices have incompatible network access methods. Therefore, we cannot directly use the existing IoT middle-ware technique to uniformly manage all devices in this system.

B. The Application

The device knowledge graph is deployed in L1 and l2 IoT middle-ware simultaneously. L1 IoT middle-ware stores only the root node of the device and the profile of the device, such as brand, model, name, etc. L2 IoT middle-ware stores the dynamic data of the device, such as the values collected by the sensor, the running state of the pump, etc. The service for accessing the knowledge graph is provided through SPARQL endpoint described in Section **??**. In the query or modification operation, L1 IoT middle-ware can access further to L2 IoT middle-ware according to the actual query and operation. In this case, we apply Easyui, JQuery and Spring Boot to build client UI and application server, which are deployed in L1 IoT middle-ware. The client and application server access the IoT knowledge graph by the SPARQL Endpoint. Fig. 4 gives the IoT knowledge graph of 15 sites currently deployed in the system.

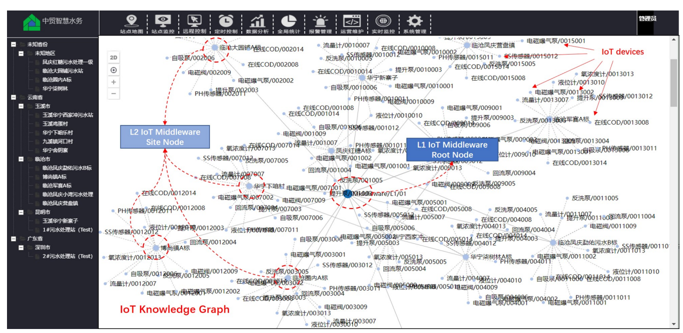

Fig. 4. The IoT knowledge graph of 15 sewage treatment stations currently deployed in the system.

C. Evaluation

In this work, we use the dedicated performance testing tool Performance Runner[3] to test the deployed IoT system. First, we set up 50 virtual users to simulate concurrency in Performance Runner. The initial number of virtual users is 10, increasing by 10 every 30 seconds. Hold for 90 seconds after adding 50 virtual users. After that, the number of virtual users began to decrease gradually, by 10 per 30 seconds, until the number of virtual users stopped at 0.

On the basis of virtual concurrent users, we choose a typical service query process for this case. First, querying a sewage treatment station root node (SEARCH-Site) in the IoT

[3]http://www.spasvo.com/Products/PerformanceRunner.asp

knowledge graph. Then, according to the site node, the status of all devices and sensors in the sewage treatment station is queried (SEARCH-Device). Finally, the sewage reflux pump is selected and the IoT knowledge graph modification operation is performed to modify the states of the reflux pump (UPDATE-Device). Fig. 5 gives the response time of IoT knowledge graph service of the sewage disposal system. Fig. 6 gives The concurrent queries per second of IoT knowledge graph service of the system.

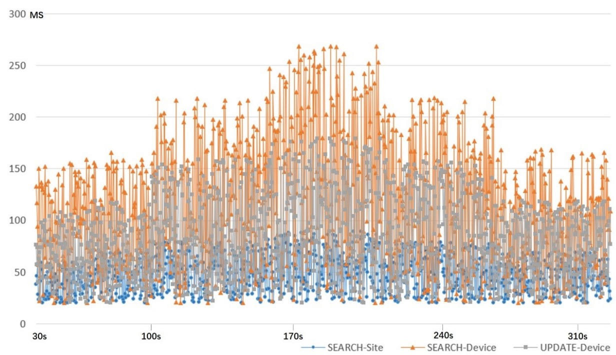

Fig. 5. The response time of IoT knowledge graph service of the sewage disposal system.

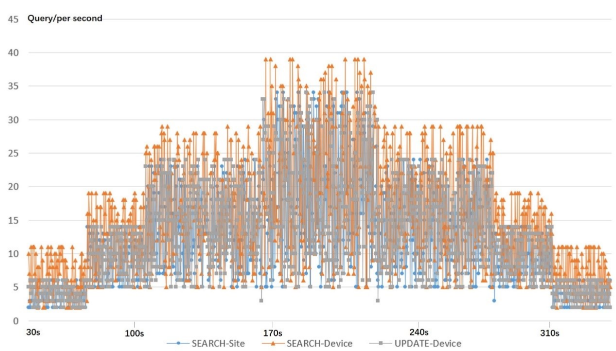

Fig. 6. The concurrent queries per second of IoT knowledge graph service of the sewage disposal system.

From Fig. 5 and 6, we can see that under 50 concurrent virtual users, the response time of three types of IoT knowledge graph service can be kept below 300 milliseconds. The overall average response time is 127 milliseconds, which meets the requirements of General Service-based application systems for response time ($<$ 200 milliseconds). For single-node query of a site (SEARCH-Site), the average response time is 43 milliseconds. For multi-node query service of devices (SEARCH-Device), the average response time is 151 milliseconds. It can be seen that the time consumed by SEARCH-Device is three times that of SEARCH-Site, which has two main reasons: (a) SEARCH-Site only needs to locate the root node of the site, does not need to go deep through the knowledge graph, thus the response time is short. (b) SEARCH-Device needs to call the L2 IoT middleware where the certain device is located (SEARCH-Site only needs to access L1 IoT middleware) that cases more node queries while the response time increases accordingly.

V. Conclusion

In this paper, a multi-layers IoT middle-ware to provide flexible IoT device access has been proposed. The proposed solution is based on IoT knowledge graph which aims to reduce the heterogeneities among IoT devices. The multi-layers middle-ware supports both WAN and field protocols that enables small IoT devices can be also accessed. The feasibility of the proposed solution, and its limited overhead, has been demonstrated through a real-life IoT project, a remote monitoring project of rural sewage treatment stations located at Yunnan Province, China. Performance evaluation demonstrates the effectiveness of the proposed solution in providing acceptable latencies in real-life IoT project and its scalability for medium-sized deployments.

References

[1] A. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, and Q. Z. Sheng, "Iot middleware: A survey on issues and enabling technologies," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 1–20, Feb 2017.

[2] D. Raggett, "The web of things: Challenges and opportunities," *Computer*, vol. 48, no. 5, pp. 26–32, May 2015.

[3] L. Yao, Q. Z. Sheng, and S. Dustdar, "Web-based management of the internet of things," *IEEE Internet Computing*, vol. 19, no. 4, pp. 60–67, July 2015.

[4] Y. Qin, Q. Z. Sheng, N. J. Falkner, S. Dustdar, H. Wang, and A. V. Vasilakos, "When things matter: A survey on data-centric internet of things," *Journal of Network and Computer Applications*, vol. 64, pp. 137 – 153, 2016.

[5] M. Eisenhauer, P. Rosengren, and P. Antolin, "A development platform for integrating wireless devices and sensors into ambient intelligence systems," in *2009 6th IEEE Annual Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops*, June 2009, pp. 1–3.

[6] K. Aberer, M. Hauswirth, and A. Salehi, "A middleware for fast and flexible sensor network deployment," in *Proceedings of the 32Nd International Conference on Very Large Data Bases*, ser. VLDB '06. VLDB Endowment, 2006, pp. 1199–1202. [Online]. Available: http://dl.acm.org/citation.cfm?id=1182635.1164243

[7] Y. Qu, L. Jie, L. Kang, Q. Shi, and Y. Dan, "Question answering over freebase via attentive RNN with similarity matrix based CNN," in *Proceedings of The 17th International Semantic Web Conference*, ser. ISWC' 18. Monterey,California, USA: Springer, 2018.

[8] Y. Zhang, H. Dai, Z. Kozareva, A. J. Smola, and L. Song, "Variational reasoning for question answering with knowledge graph," in *Proceedings of The Thirty-Second AAAI Conference on Artificial Intelligence*, ser. AAAI'18. New Orleans, Louisiana, USA: AAAI Press, 2018.

[9] K. M. Annervaz, S. B. R. Chowdhury, and A. Dukkipati, "Learning beyond datasets: Knowledge graph augmented neural networks for natural language processing." in *Proceedings of The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, ser. NAACL HLT' 18, New Orleans, Louisiana, USA, 2018.

[10] D. H. E Cambria, S Poria and K. Kwok, "Senticnet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings." in *Proceedings of The Thirty-Second AAAI Conference on Artificial Intelligence*, ser. AAAI'18. New Orleans, Louisiana, USA: AAAI Press, 2018.

[11] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," in *Proceedings of The Thirty-Second AAAI Conference on Artificial Intelligence*, ser. AAAI'18. New Orleans, Louisiana, USA: AAAI Press, 2018.

[12] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "Dbpedia a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, Jan. 2015.

[13] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A core of semantic knowledge," in *Proceedings of the 16th International Conference on World Wide Web*, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 697–706.

[14] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka, Jr., and T. M. Mitchell, "Coupled semi-supervised learning for information extraction," in *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, ser. WSDM '10. New York, NY, USA: ACM, 2010, pp. 101–110.

[15] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: ACM, 2014, pp. 601–610.