2016 IEEE 8th International Conference on Cloud Computing Technology and Science

A Pub/Sub-Based Fog Computing Architecture for Internet-of-Vehicles

Sejin Chun, Sangjin Shin, Seungmin Seo, Sungkwang Eom, Jooik Jung, and Kyong-Ho Lee
Department of Computer Science
Yonsei University
Seoul, Republic of Korea
{sjchun, sjshin, smseo, skeom, jijung}@icl.yonsei.ac.kr, khlee89@yonsei.ac.kr

***Abstract*—Fog computing is a promising paradigm in terms of extending cloud computing to an edge network. In a broad sense, fog computing in Internet-of-Vehicles(IoV) provides low-latency services since fog nodes are closely located with moving cars and are locally distributed. In this paper, we propose a fog computing architecture based on a publish/subscribe model. After that, we describe a traffic congestion control scenario using a smart traffic light system which operates on top of the proposed architecture. Furthermore, we propose an upper-level domain ontology in order to enhance the expressivity of knowledge and describe a variety of semantic properties which interlink spatial information in IoV. Finally, we present an active rule where supports the exchange of event-driven messages between publishing and subscribing fog nodes.**

Keywords—Internet-of-Vehicles; Internet-of-Cars; Fog computing; OWL; Semantic Web; Pub/Sub system;

I. INTRODUCTION

Internet-of-Vehicles (IoV)[1] is a term representing a new era of Vehicle Ad hoc networks (VANETs). VANETs[2] encourage all participating vehicles to connect themselves to wireless routers or mobile nodes. However, VANETs only cover a small range of networks which contain a limited number of connected cars and restrictions in mobility. Therefore, VANETs cannot support global scale services or applications.

In contrast to VANETs, IoV is based on a large scale network which supports services for big cities and a whole country. Based on such networks, the vision of IoV is to enable diverse interoperations among the participating vehicles and infrastructures. Some of Internet-of-Things(IoT) applications such as a smart city, process distributed endpoints at a private data center by using "Pay-as-you-go" as a centralized Cloud computing model. However, the Cloud computing model cannot respond to delay-sensitive services efficiently such as car accidents and etc.

Fog computing[3], on the other hand, resolves issues such as high mobility and spatial distribution of vehicles in IoV, as well as location cognition and low-latency requirements. In general, fog computing has a distributed structure which extends a Cloud to the edges of a network. Accordingly, it can support low-latency services, geo-distributed applications, and rapid mobile applications. Thus, fog computing provides a seamless and optimized connectivity and interoperability, e.g., WiFi, 3G, LTE, and so on.

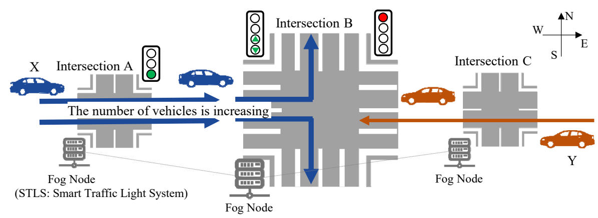

Fig. 1. A scenario of a traffic congestion in IoV

Recently, many approaches have been suggested to build components for managing resources in fog computing [5,7,8,9]. Their components cope with major issues[3] such as wide geographic distribution, low latency, and so on. However, the knowledge exchange in those methods limits scalability and interoperability. This means that a rich and common data model is required to express and propagate the knowledge at the semantic level. As a recent research of applying Semantic Web technologies[15] to distributed nodes, the authors of [4] propose a semantic-based federated node architecture for IoT. Although their architecture interacts with nodes efficiently, it requires an explicit request on the distributed knowledge, since their data model is a query-driven.

This paper proposes a fog computing architecture where IoV knowledge is semantically represented, published and subscribed. As a motivating example, we first describe autonomous STL(Smart Traffic Light) systems scenario where multiple vehicles and fog nodes cooperate as a response to traffic congestion. Based on this scenario, we propose an abstract pub/sub architecture where semantic knowledge is processed. Moreover, we construct an upper-level ontology in order to process raw data in order to acquire high-level knowledge. The ontology supports various types of events and includes relative relationships of spatial information such as traffic paths. Finally, we present an active rule that expresses to either publish or subscribe semantic knowledge among fog nodes.

978-1-5090-1445-3/16/$31.00 ©2016 IEEE (CloudCom'16)

2330-2186/16 $31.00 © 2016 IEEE
DOI 10.1109/CloudCom.2016.26

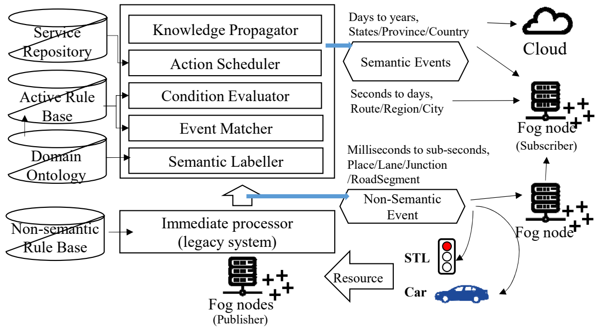

Fig. 2. The proposed architecture in a fog computing

II. Motivating Scenario

Figure 1 shows a motivating scenario where autonomous STL systems coordinate themselves to alleviate road congestion problems. Each STL system monitors various types of vehicular data streams, e.g., a speed, direction, and location of a car. The STL system can also control the traffic lights according to certain events, e.g., a car accident, traffic congestion, and so on. In this scenario, each STL system is represented as a fog node, deployed at every intersection, and connected with each other via a field area network such as WiFi, 3G/4G, and LTE. To offer a low-latency service in time, a fog node collects the required vehicular data streams from the vehicles that are in proximity.

In Figure 1, intersection B is located between intersection A and intersection C. Moreover, X indicates the traffic flow in the direction of A to B, and Y represents the traffic flow from C to B. Suppose that the number of moving cars from A to B is multiplying during a certain time period(e.g., a rush hour), and the number of moving cars from C to B is relatively less than the former. In this case, both intersections A and B must have their green signals in the east direction extended, whereas intersections B and C must have their red signals in the west direction. As a consequence, the STL systems can solve the traffic congestion problem on the road by cooperating with each other.

From the above motivating scenario, we raise three major issues: 1) how to share information between the fog nodes consistently and explicitly, 2) how to provide rich descriptions of events as they occur at the fog level, and 3) how to plan out the cooperation between the fog nodes that are interested in a particular event.

III. The Proposed Approach

A. The proposed Architecture

Figure 2 shows our fog computing architecture which is based on a publish/subscribe model. The proposed architecture satisfies the requirements about a low-latency service due to the geo-distribution of fog nodes. Our fog nodes create high-level knowledge and share them with each other consistently and explicitly using ontology. The interactions between the fog nodes are based on a topic-based publish/subscribe model[6], where a publisher issues a message with a 'topic' and a subscriber receives the message if he/she is interested in the topic. In our approach, a publishing fog node is responsible for defining the class of an event pattern to which a subscribing fog node can subscribe. Therefore, the fog nodes can work together and respond to the same event that they are subscribed to. This also means that even though an event can be propagated to all fog nodes, only the ones that are subscribing the event can accept and receive information.

Specifically, each fog node continuously collects data transferred from cars, STLs, and so on. For example, a red or a green signal in a traffic light, the observed values of both speed and direction of a vehicle, the status of a damaged car can be part of the data mentioned previously. The gathered information undergo two layered processes of acquisitioning high-level knowledge.

The first layer, the immediate processor, captures various event patterns from the gathered information. For example, when a fog node receives the damage analysis information of a crashed car, it tries to identify whether a car accident has occurred or not with the pre-defined rules. These rules are used to infer a car accident event from the damage analysis information. Once the event is defined, the fog node notifies the inferred information to other fog nodes or vehicles that are in the proximity of the car accident. In general, such information may be generated in a unit of milliseconds to sub-seconds and have their transmission range of a small region such as a place, lane, junction, road segment, and so on.

The second layer contains several components which are used to derive high-level knowledge from the information processed by the first layer. To efficiently exchange knowledge among fog nodes, we utilize Semantic Web technologies. An ontology such as Web Ontology Language(OWL)[14] provides homogeneous and scalable means of accessing objects in IoV. Meanwhile, a domain ontology is used to define knowledge in a specific system as a set of concepts and relationships among them.

Using the domain ontology, the semantic labeler wraps the resources with semantic information. In order to effectively exchange such semantically enriched resources among fog nodes, we use an event-driven architecture based on active rules. Thus, we define 'semantic event' terminology as a significant change in the resources, and refer to a complex trend of the semantic event as 'semantic event pattern'. An active rule defines which action to take in response to a specific semantic event, using the ontology. An event matcher captures semantic events by evaluating the resources with a set of active rules. A condition evaluator checks whether a specific action can do or not when a specific event is detected. An action scheduler manages a sequence of actions. In order to share the captured semantic events, the knowledge propagator forwards them to other fog nodes or the cloud that are subscribing to the events.

978-1-5090-1445-3/16/$31.00 ©2016 IEEE (CloudCom'16)

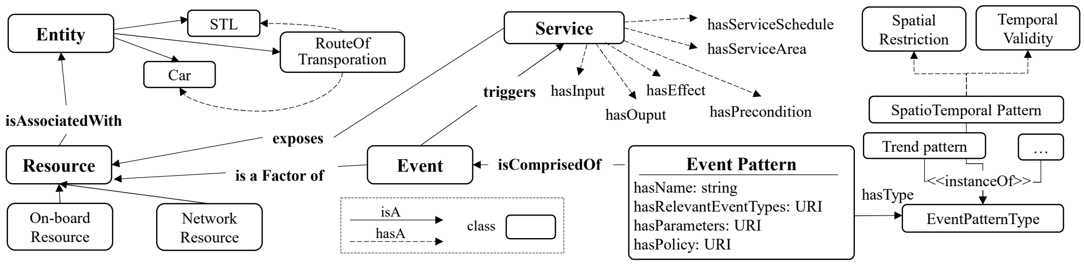

Fig. 3. IoV domain ontology

Some of the semantic events, which are delivered to the cloud, may be generated in a unit of days to years, and have their transmission range of a large region such as a state, province, country, and so on. On the other hand, the events which are forwarded to other fog nodes, may be generated in a unit of seconds to days and have their transmission range of a medium sized region such as a route, region, city, and so on.

B. Domain ontology

Figure 3 illustrates the proposed domain ontology for IoV. The ontology consists of classes, individuals, relations, and attributes. A class describes concepts in IoV applications, an individual describes an instance of a class, an attribute indicates features and characteristics of a class, and a relation describes a relationship either between classes or individuals. The proposed ontology allows easier integration of ontologies between both IoT and IoV applications by reusing the core classes as concepts of an IoT ontology[12].

Above all, we extend the ontology to include an event concept as a contextual change within an IoV(or IoT) environment while utilizing the concepts such as an entity, resource, and service. Due to the limitation of expressing various types of events, we define a notion of the event and its relationships among other classes, as discussed in our previous work[13].

In the proposed ontology, Entity class indicates objects in real world. For example, we define IoV entities as all vehicles and STLs, and traffic guidance, as well as a route of transportation(thoroughfare), e.g., land, intersection, and so on. We adopt the ontology of [10] that provides spatial concepts and relations among them over a network of vehicles, e.g., an intersection class is a subclass of a junction class. Resource class, which is associated with an entity, indicates information for controlling and managing the entity. Specifically, the resource class is classified as an on-device resource or a network resource. An on-device resource indicates information that is hosted on a device such as a vehicle. On the other hand, a network resource indicates information that is always accessible via a network, e.g., a cloud or a fog node.

Service class, which exposes resources, provides a well-defined and standardized interface that offers functionalities for interacting with an entity. The functionalities of a service are expressed using the properties of hasInput, hasOutput, hasPrecondition, and hasEffect. Also, we specify a region and a schedule of a service using hasServiceArea and hasServiceScehdule properties, respectively. Event class, which triggers a service, represents a status of a significant change in IoV. For example, an over-speed event is defined as a car driving over the speed limit. The fog node, which observes the car, detects the event and sends a warning message to the car by triggering an alert service.

To look for a collection of events like a green wave of STL, we define an event pattern as a template for specifying a combination of two or more individual events. An event pattern expresses a trend of constituent events. In detail, hasName property indicates the name of an event pattern and hasRelevantEventTypes property indicates a set of individual event types relevant to an event pattern. hasParameter property specifies additional variables for a function of a matching pattern and hasPolicy property determines operations of pattern matching, e.g., an evaluation policy for producing the matching results. The eventType property indicates the type of an event pattern such as a trend pattern, spatial-temporal pattern, and so on. For example, in the case of a trend pattern, an instance of an increasing pattern is satisfied if the number of cars increases monotonically.

To analyze dynamic IoV data over time depending on geo-distribution of fog nodes, we present a spatial-temporal event pattern. This pattern represents time series of events and a spatial trend over time. For example, patterns of moving in a constant direction, stationary, and moving toward can be part of spatial-temporal sub-patterns. These patterns contain both spatial restrictions and temporal validity classes. The former specifies constraints of spatial information of relevant events, e.g., a specific intersection. The latter specifies the valid duration for the participating events, e.g., duration between start- and end-dates of events. For example, a stationary pattern instance has ‘A intersection’ for the spatial restriction and ‘one hour’ for the temporal validity.

ON Event (?evt) ^ hasLane(?evt, ?l) ^ isCongested(?l, true) → CongestedLane(?evt)
LET ?nodes := select neighboring nodes of subscribing CongestedLane event
IF hasCongestionLevel(?l, ?level) ^ ?level > 60
DO publish (?evt, ?nodes)

Listing 1. Example of an active rule at publishing node

ON newEvent(?evt) ^ hasCongestedLaneEvent(?evt, ?con)
LET ?light := select a traffic light controlling car movements
IF true
DO controlTrafficLight(?light)

Listing 2. Example of an active rule at subscribing node

978-1-5090-1445-3/16/$31.00 ©2016 IEEE (CloudCom'16)

C. Active Rules for Event-Condtion-Action

To specify a response behavior to an event and its patterns, we adopt an active rule[11]. These rules use a well-defined formalization and are based on a structure of Event-Condition-Action rules as follows:

ON event (**LET** domain-specific content) **IF** condition **DO** action

The content of an active rule provides intuitively a formal meaning: when an event(or event pattern) occurs, evaluate a condition, and if the condition is satisfied then execute an action. Each statement in the content is described using the domain ontology proposed earlier. In addition, a domain-specific content in LET clause indicates local variables and constants for defining active rules active, e.g., subsets of a dataset in a fog node can be obtained via SPARQL.

Listing 1 and 2 describe examples of active rules for the motivating scenario introduced earlier. In the Listing 1, there is an active rule in the publishing fog node. Specifically, the fog node identifies CongestedLane event as a type of an event which has a lane that is congested. In the LET clause, the fog node selects neighboring nodes that are subscribing the CongestedLane event. If the lane's congestion level is 60 or higher, the CongestedLane event is notified to the neighboring nodes. In the Listing 2, there is an active rule in a fog node that subscribes to the CongestedLane event. In the ON clause, the CongestedLane is defined as the event to which the fog node is subscribing. To control the car movements, the traffic lights are retrieved in the LET clause. Lastly, the fog node sends a message for controlling the traffic lights.

The publishers and subscribers in fog nodes cooperate with each other easily and reactively, by defining active rules and exchanging events that they are interested in. The cloud can monitor the events generated from distributed fogs by defining the events as active rules. For instance, a cloud-based analytic service subscribes to various types of car accident events and receives information about them whenever they occurs.

IV. Conclusion and Future work

The paper introduced a pub/sub-based fog computing architecture for IoV. Firstly, we drew three major issues from a scenario of autonomous STL system. Secondly, we presented a fog computing architecture based on a publish/subscribe model. In the proposed architecture, fog nodes declare complex events and share them with other nodes using our IoV domain ontology. Lastly, we presented how to publish semantic events and subscribe to the events occurred using active rules. Our future work will be dedicated to extending our architecture and performing various experiments using real-world IoV datasets.

Acknowledgment

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No. R7120-16-1011, Cloud-based Service Platform IoV Data Storage and Analysis).

References

[1] M. Gerla, E-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *Proc. IEEE World Forum Internet Things (WF-IoT)*, pp. 241-246, 2014

[2] A. Dua, N. Kumar, and S. Bawa, "A systematic review on routing protocols for Vehicular Ad Hoc Networks," *Veh. Commu.*, vol. 1, no.1, pp. 33-52, 2014.

[3] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," *Big Data and Internet of Things: A Roadmap for Smart Environments, Springer, Studies in Computational Intelligence,* vol. 546, pp. 169-186, 2014

[4] S. De, B. Christophe, and K. Moessner, "Semantic enablers for dynamic digital–physical object associations in a federated node architecture for the Internet of Things," *Ad Hoc Networks*, vol. 18, pp. 102-120, 2014

[5] S. K. Datta, C. Bonnet, and J. Haerri, "Fog Computing architecture to enable consumer centric Internet of Things services," in *Proc. IEEE Int. Symp. On Consumer Electronics (ICSE)*, pp. 1-2, 2015

[6] P. T. Eugster, P. A. Felber, R. Guerraoui, and A-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys (CSUR)*, vol. 35, no.2, pp. 114-131, 2003

[7] Y. Teranishi, R. Banno, and T. Akiyama, "Scalable and locality-aware distributed topic-based pub/sub messaging for IoT," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, pp. 1-7, 2015

[8] A. Mtibaa, K. Harras, and H. Alnuweiri, "Friend or foe? Detecting and osolating malicious nodes in mobile edge computing platforms," in *Proc. IEEE Int. Conf. on Cloud Computing Technology and Science (CloudCom)*, pp. 42-49, 2015

[9] Y. Sun, X. Qiao, B. Cheng, and J. Chen, "A low-delay, lightweight publish/subscribe architecture for delay-sensitive IoT services," in *Proc. IEEE Int. Conf. on Web services (ICWS)*, pp. 179-186, 2013

[10] L. Zhao, R. Ichise, S. Mita, and Y. Sasaki, "Core ontologies for safe autonomous driving," in *Proc. Int. Semantic Web Conference Posters & Demonstrations Track*, 2015

[11] W. May, J. J. Alferes, and R. Amador, "Active rules in the Semantic Web: Dealing with language heterogeneity," in *Proc. Int. Workshop on Rules and Rule Markup Languages for the Semantic Web*, Springer Berlin Heidelberg, pp. 30-44, 2005

[12] S. Chun, J. Jung, X. Jin, G. Cho, J. Shin, and K-H. Lee, "Semantic URI-based event-driven physical mashup," in *Proc. IEEE World Forum on Internet Things (WF-IoT)*, pp. 195-196, 2014.

[13] S. Chun, S. Seo, B. Oh, and K-H. Lee, "Semantic description, discovery and integration for the Internet of Things," in *Proc. IEEE Int. Conf. on Semantic Computing (ICSC)*, pp. 272-275, 2015.

[14] https://www.w3.org/OWL/

[15] https://www.w3.org/standards/semanticweb/

978-1-5090-1445-3/16/$31.00 ©2016 IEEE (CloudCom'16)