Enabling Edge Analytics of IoT Data: the Case of LoRaWAN

Hong-Linh Truong
Faculty of Informatics, TU Wien, Austria
E-mail: hong-linh.truong@tuwien.ac.at

***Abstract*—LoRaWAN is a promising network solution for various application domains, especially in developing countries. While its network architecture is highly distributed, the network architecture aims at aggregating data into a centralized location, mainly the cloud-based data center. With such an architecture, we can bring data from distributed sensing sources to centralized cloud-based data and analytics services, however, it does not foster edge analytics atop LoRaWAN, which is highly suitable for scenarios in developing countries due to connectivity and cost constraints. We outline a conceptual architecture and design augmenting LoRaWAN software architecture for edge analytics. In this paper we develop techniques for edge analytics working with LoRaWAN core network functions. We present key principles of shared data for edge analytics with LoRaWAN. Based on that, we describe main services, their interactions and data models for connecting edge analytics to LoRaWAN infrastructures. We present our prototype to illustrate our implementation.**

I. Introduction

The new LoRaWAN technology [1] has fostered interesting Internet of Things (IoT) deployments and applications. Not surprisingly, many LoRaWAN connectivity platforms - which help bringing data from LoRaWAN devices to appropriate applications - are currently being developed by industries and researchers. To date, we have seen two major infrastructural deployment models of LoRaWAN: (1) big providers layout dedicated core LoRaWAN infrastructures (e.g., gateways) and cloud connectivity platforms for different applications; and (2) public, community gateways and connectivity platforms enable shared infrastructures of LoRaWAN. An example of the first model is a telco, as an infrastructure provider, offering a network of LoRaWAN gateways and cloud services, while a government agency, as an application customer, deploys LoRaWAN devices sensing city environments and develops applications monitoring the environments. Examples of the second model are community gateways, such as The Things Network (TTN) [2], where anyone can provide his/her devices and gateways and develop his/her applications processing the device's data. Such deployment models are also tightly associated with the underlying business models, such as a pay-per-use model for LoRaWAN network connectivity and data management in the first case, or free services for all in the case of TTN. These deployment and business models strongly influence the design and development of LoRaWAN connectivity platforms, affecting the way of how applications could utilize data from LoRaWAN devices.

Independent from these deployment models, in the design of existing platforms for LoraWAN, application data processing is currently very much centralized: we push all the data from devices to the Application Server in the cloud, although the LoRaWAN networks of devices and gateways are distributed in sparse geography. Application customers of LoraWAN infrastructures own their devices and process their data in the cloud. It is difficult for other customers to utilize the application data from the devices that do not belong to the customers, unless one makes the data available at the cloud. This way of design and deployment is not suitable for many places, e.g. in developing countries, where network connectivity is unreliable and accessing cloud services is expensive in terms of costs and performance [3]. The current model does not exploit the potentials for supporting analytics in the edge in LoRaWAN, based on the edge computing models [4], [5], [6], [7].

In this paper, motivated by technical and business requirements from developing countries, we need to support multiple providers for different types of LoRaWAN-based services (device-as-a-service, gateways-as-a-service, and data-as-a-service). In such countries, we are faced with several issues that require feasible solutions for IoT infrastructures to solve crucial problems [3]. To this end, we propose to augment existing LoRaWAN architectural software designs to allow application data sharing within the core architecture of LoRaWAN through customized and loose-coupling between LoRaWAN Network Server and our middleware for enabling edge analytics for multiple application customers, who might or might not deploy/own LoRaWAN devices. We contribute novel techniques, implemented in our proof-of-concept IoTRACE prototype, which allow multiple data subscribers deploy their analytics applications in the edge by utilizing shared applications from the device owners and also demonstrate how one can combine edge analytics with cloud analytics for such application data.

The rest of this paper is organized as follows: Section II discusses the background and motivation. Section III gives an overview of IoTRACE. We elaborate important features for edge analytics in Section IV. Experiments are shown in Section V. Related work is presented in Section VI, while Section VII concludes the paper and outlines our future work.

II. Motivation

Detailed in [1], the key elements of a LoRaWAN network are the *Gateway* and the *Network Server*, whereas Devices are usually provided by LoRaWAN *Device Provider*. *Application Server* is the external service subscribed/deployed by the

Application Device Provider to obtain *application data* sent from the Device. To enable data aggregation and analytics atop LoRaWAN infrastructures, several cloud-based connectivity platforms have been developed for the Application Server and core features of LoRaWAN (see Section VI). They mainly enable LoRaWAN Network Server to push data into the Cloud using well-known IP-based protocols, like MQTT, AMQP, and HTTP. Currently, it is quite straightforward to connect the Network Server to existing cloud-based systems to enable the transmission of application data from Devices. With the current software design, data analytics at the edge [4], [7], close to the Devices and Network Servers, is not well supported. First, the Network Server mainly has static configuration of the back-end cloud-based Application Server; leading to the difficulty of performing data analytics close to the Network Server. Second, the infrastructural elements (Devices, Gateways, Network Servers) might come from multiple providers, requiring complex interactions for enabling edge analytics.

Several papers have discussed the benefit of edge analytics [4], [8]. We advocate for edge analytics with LoRaWAN due to two main reasons: (i) LoRaWAN devices are quite suitable for monitoring of environments and agriculture with low costs and weak network connectivity as key constraints, especially for monitoring environments in large sparse geography, like in developing countries, and (ii) shared data from LoRaWAN devices should be enabled at the edge due to the deployment cost constraints. We detail our motivation in the following:

1) Application scenarios: Monitoring salinity in waters in the Mekong delta in Vietnam is very important as the farms rely on quality of water. However, farmers do not want to setup LoRaWAN devices (and sensors), instead, they want to buy the data; they are the data users. On the other the hand, no single company will deploy a vast LoRaWAN infrastructure due to the high cost of LoRaWAN infrastructures and services. In the context of Vietnam, therefore, we have the following providers advocating infrastructure-as-a-service principles for LoRaWAN deployments:

- Device Providers can be some farmers, when they have big farms, other companies and the local government agency, but application data from Devices should be utilized (with prices) by farmers and other stakeholders.
- Infrastructure Providers should also enable analytics applications (e.g., for farmers) to utilize the data without connecting the cloud to save costs. Certain users, like the government agency, would need edge and cloud analytics to have a better view on the salinity problems to develop policies and agricultural infrastructures.

Similar situations can be also seen in fish farms in open seawater bays along the center of Vietnam. In these seawater bays, the water monitoring should provide various parameters about quality of water to detect potential problems for fishes and lobsters. Several farms share and lease the spaces in the bay, however, any bay is too vast for them to deploy sensors and networks. For the above-mentioned scenario, in addition to the case of having different providers, we see that it is unnecessary to focus on the centralized model of LoRaWAN connectivity platforms which push all the data to the cloud. The reason is that connectivity to the cloud is a big problem in such scenarios, especially from the Gateways to the Network Servers (based on IP). Furthermore, there is no need to push the data to the cloud and then obtain the analytics from the Cloud, as most requests are from farmers.

2) Key augmented features: There are different analytics applications which rely on different data pipelines analyzing data from Devices. Therefore, while we still need core features of LoRaWAN like Devices, Gateways, and Network Servers, we do not just take the data at the clouds and apply data pipelines for the data at the centralized place. Instead, data pipelines for applications will need to process LoRaWAN application data at various points in the LoRaWAN infrastructures. A data pipeline for an analytics application would need to interact with, e.g., device-as-a-service, gateway-as-a-service, and Network Servers. Furthermore, both edge and cloud-based analytics are desired.

III. IOTRACE ARCHITECTURE

Figure 1 describes the overall architecture of our conceptual software architecture, namely IoTRACE. We focus on two main types of building blocks: *Network Server* and *IoT Data Hub* for both edge and cloud analytics. These building blocks include software services (and corresponding computing infrastructures) handle data from LoRaWAN Devices and analytics of the data. Other building blocks, *Service Information Management* and *Monitoring and Billing*, provide other core features for information management, billing and monitoring.

A. Service Information Management

Service Information Management (IMS) manages Providers, Devices, Gateways, Network Servers, Data Hubs, and Data Subscribers, and provides functions for managing and verifying them. To allow edge analytics, service information must be available for middleware components (of IoTRACE) deployed in Gateways, Network Servers and Data Hubs blocks. Especially, IMS must manage application data subscribers (and their corresponding Data Hubs) and make such information available for the Network Server to enable distributed data delivery at the edge. Our IoTRACE supports multiple stakeholders and data contracts for IoT. First, we have different Device and Gateway Providers which deploy their Devices and Gateways. Second, we will have different Application Providers, which usually own devices (thus they can be Device Provider and Data Provider when they share the application data). Finally, we will also have Data Subscribers, which buy data from Devices and obtain data for specific applications; the Data Subscribers receive the subscribed data through dedicate/on-demand IoT Data Hubs.

B. Network Server

Standard features of the Network Server in LoRaWAN are for accepting LoRaWAN messages, de-duplicating messages, and ingesting messages [1] into Data Hubs for applications. For enabling edge analytics, first, in our software design,

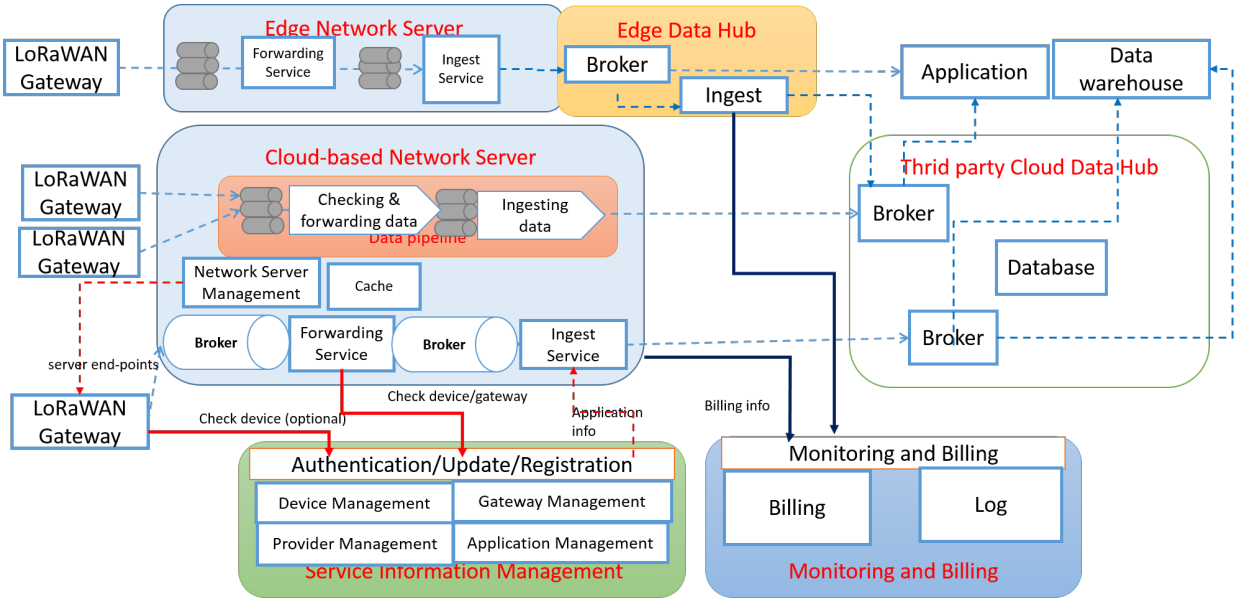

Fig. 1. Overall architecture of IoTRACE

depending on the configuration, we have Network Server for centralized Clouds and for edge servers enabling edge analytics. Second, its features are highly configurable and the target of application data (Data Hubs) must be programmable, e.g., through software-defined capabilities, to allow us to push data to the right analytics application in the edge. Conceptually, the LoRaWAN application data moved to specific applications are handled by tool pipelines built atop these features. We will discuss it in detail in Section IV.

C. IoT Data Hub

A Data Hub is where Data Subscribers receive the application data from Devices. It provides the data to analytics applications, which might subscribe different types of data. (Third party) Data Hubs support us to provide messages to the applications typically through a large-scale messaging broker service. We consider both third-party Data Hubs in the cloud and in the edge system. In our design, they can be from public centralized cloud providers and private clouds and edge-based Data Hub to enable edge analytics. Since, cost and performance are crucial issues, the architecture has to provide various adapters for different types of Data Hubs.

D. Monitoring and Billing

To allow edge analytics, we need to enable connections from Gateways, Network Servers, and edge analytics applications to Monitoring and Billing Services. This is different from architectures for centralized clouds, when billing can be done at a central place. This feature is out of the scope of this paper.

IV. AUGMENTING CENTRALIZED DATA AGGREGATION WITH EDGE ANALYTICS

A. Stakeholder and Edge Data Sharing Principles

As said, we have to support edge data sharing principles in terms of allowing non-owner of devices to access device's data. Therefore, one of the main important points is to have the right model to manage different stakeholders. There are many entities to be managed and information about such entities are often accessed for other actions, thus a highly-scalable entity management based on NoSQL database is needed. However, in this paper, the key technical challenge for us is to link different types of information and to allow Network Servers and other components to use the information to dynamically configure data ingestion for edge analytics. We distinguish the following important roles:

- Provider: provides infrastructures or data. In our model we have: (i) Infrastructure Provider for providing Gateways, Network Server, LoRaWAN Cloud connectivity platforms, (ii) Application Provider for providing Devices and data produced by devices. The Application Provider is also the Application Data Provider in this paper.
- DataSubscriber: subscribes application data provided by the Application Provider (also the Application Data Provider).

Figure 2 shows simplified relationships among various types of data for enabling edge analytics. Certain information elements, like, `NetworkSessionKey`, `ApplicationSessionKey`, `Device`, `Gateway`, `NetworkServer`, are known in most LoRaWAN infrastructures and connectivity platforms; they keep basic information about entities to ensure registration, authentication and information security. In our work, we focus on extending them with necessity for enabling edge analytics:

- `DataSubscriber`: represents different kinds of customers who want to receive data (for analytics), whether they own the devices or not.
- `DataContract`: describes information that data subscribers want and how to deliver the data.

- `DataHub`: describes information about Data Hubs through the data should be delivered. `Data Topic` will be associated with Data Hubs and Contracts, enabling `AnalyticsApplication` to receive the right data.

Another extension is about interactions among services. For example, typically only the Application Server knows the `ApplicationSessionKey` in order to decrypt the application data from LoRa messages (before sharing the data to other customers). To enable edge analytics and multiple data subscribers, the Application (Data) Provider has to delegate `ApplicaionSessionKeys` to instances of application data extraction components in the edge that implement certain features of the Application Server. This requires us to provide *Edge Application Data Extractor* (see Section IV-B) that will be invoked when the Application Provider has multiple DataSubscribers in the edge. Another issue is that, when a DataSubscriber invokes its edge analytics application, the DataSubscriber mostly specifies the application identifier which links to the `DataContract`. Using the application identifier, we must find out relevant device addresses and `ApplicationSessionKeys` in order to decode the application data.

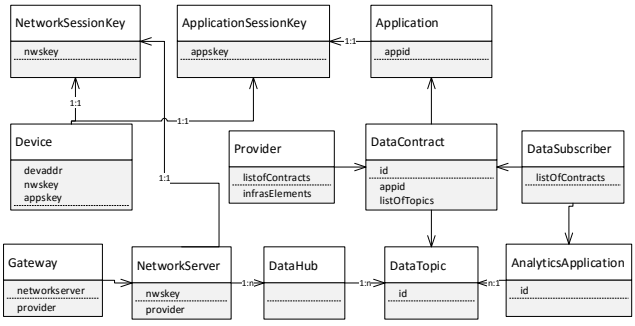

Fig. 2. Simplified information model

B. Dynamic Data Ingestion in Network Servers

We focus on changing the ingestion of messages consisting of application data to Data Hubs and the support of `AnalyticsApplication` in the edge. It means we modify the current way of Network Server that just simply forwards the (encrypted) data from the Devices to pre-defined Application Server. Figure 3 outlines main components and interactions within our proposed IoTRACE Network Server, including typical Network Server functions and edge data analytics functions.

1) Ingestion Mechanism: Key changes in ingesting components in the Network Server are related to (i) ingesting data to the right Data Hubs, based on data contracts of edge analytics applications and (ii) reconfiguring Network Server for runtime edge applications. Shown in Figure 3, we distinguish functions of typical Network Server and functions for edge data management and analytics. We see clearly three different types of roles: Infrastructure Provider for Network Server and Connectivity Platform, the Data Provider (for devices and their data), and the Data Subscriber. The Network Server performs its normal functions (checking and filtering data). To enable edge analytics the Network Server must forward LoRa messages to the right Data Hubs in the edge and the messages must be decrypted so that applications of Data Subscribers can utilize the data. These needs should not be done within the Network Server function, because we need to acquire appropriate `ApplicationSessionKey` for decrypting messages while Network Server cannot have these keys from multiple owners of devices providers. Since the Network Server only knows the device address and the Data Subscribers should not know the `ApplicationSessionKey`, we introduce a lightweight *ApplicationDataExtractor* to extract the application data from messages for DataSubscribers; this Extractor has to be executed under the role of the owners of the data produced by the Devices. One could see that this Extractor performs certain limited functions of the Application Server in the LoRaWAN architecture at the edge and each instance of the Extractor would serve for one Data Provider.

2) Application Data Extractors: In Figure 3, the *Ingest Service* will push messages to the *EdgeAppDataExtractor*. Using the device address[1], within the Extractor, we obtain external information about `DataContract` and `DataSubscriber` (`Device` is linked to `ApplicationSessionKey` to `Application` to `DataContract`). From the information, we determine if application data extracted from a message sent by a device will be shared or not, and then we decide to push the data into single Data Hubs or not. For example, we could obtain the list of *DataSubscribers* associated devices in a JSON-based info:

```
{
      "devaddr":[3,2,1,0],
      "datasubscribers":["test","test2","test3"]
}
```

Based on that, it determines the right Data Hub and corresponding `DataTopic`. Using `ApplicationSessionKey` it decrypts data and push to the right hub. Note that `EdgeAppDataExtractor` is executed under the *DataProvider* role, which owns devices and has ApplicationSessionKey, although from the software perspective it can be run in the same or different computing platform with the Network Server. As an example, the following code shows features of Network Server, which forwards messages with application data encrypted to input queues of `EdgeAppDataExtractor`. Each extractor, after finishing the application data decryption, forwards the application data to corresponding Data Hubs:

```
#Network Server moves data to input queues of extractor
for subscriber in subscribers:
   extractor_topic = "dataextractor/" + subscriber + "/
        iotrace"
   extractorqueue_conn.publish(extractor_topic, app_data)
#.....
# Extractor
def on_edge_data(measurement):
    datacontract_topic = "application/" + app_id + "/
         iotrace"
    datahub_conn.publish(datacontract_topic, measurement)
```

[1]E.g., the device address *devadd* can be obtained from the header of the message $fhrd$ as `devaddr = fhdr.get_devaddr()`

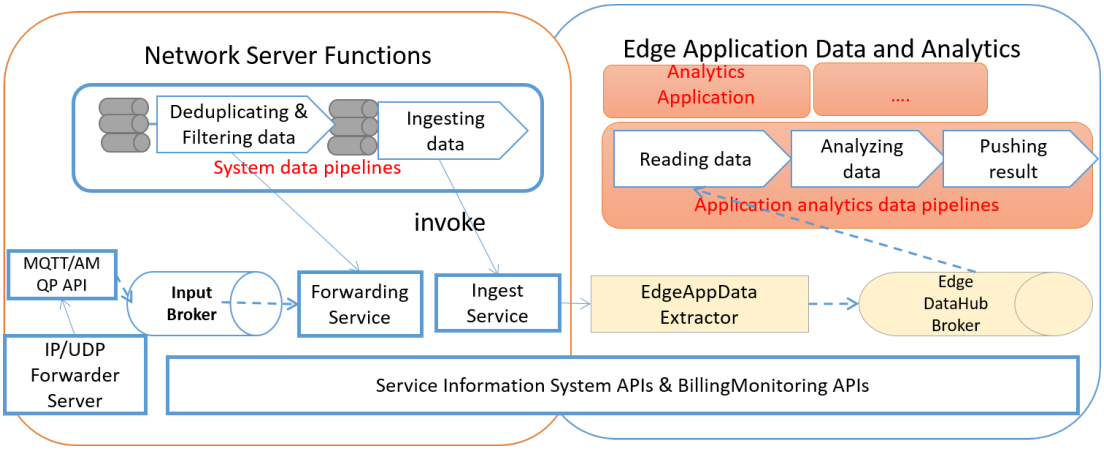

Fig. 3. Main components of the IoTRACE Network Server integrating network server and analytics functions

3) Data Hubs for Edge and Cloud: Our architecture just utilizes (third party) Data Hubs, e.g., based on MQTT, AMQP, Kafka, or Google Pub/Sub, through various connectors. The key technical challenge for us is to manage the connectivity to the Data Hubs for Data Subscribers and to deploy edge Data Hubs for Data Subscribers, if needed. Currently, for the Data Hubs at the edge, we use MQTT and AMQP.

V. Prototype and Edge Analytics

A. Prototype

To demonstrate our software design, we have developed a prototype. For the LoRaWAN Devices and sensors, we emulate them. Our emulated devices read real-world sensor data from files and using the LoRaWAN python library[2] to create LoRa packets. We emulated two cases: lora packets are sent to Gateways via message queues and to Packet Forwarder using the LoRa-Gateway-Bridge[3]. We have developed our designed LoRaWAN Network Server using python and MQTT, enabling features of LoRaWAN architectures to support simple push data protocols for edge analytics.

B. Analytics

Since our paper is mainly about software component changes and interactions, to demonstrate our concept, we have not developed a full prototype of the whole system. Our goal is to incorporate our concept with existing LoRaWAN platforms. Therefore, we do not show here performance analysis or large-scale deployment but illustrate examples of how we could enable edge analytics.

1) Sampling data: We used real-world data from our Base Transceiver Station monitoring system [4]. Sample of output data is:

```
{"threshold": "56", "start_time": "2017-06-08 00:01:43",
  "id": "8098276", "alarm_id": "312", "parameter_id":
       "141",
  "end_time": "2017-06-08 00:01:44", "station_id":
       "1161115043",
"value": "56.5"}
```

[2]https://github.com/jeroennijhof/LoRaWAN

[3]https://docs.loraserver.io/lora-gateway-bridge

[4]The system is developed using Raspberry PI and 3G to monitor BTS in a large telco company in Vietnam

2) Edge Analytics: For only edge analytics, one can use different techniques to subscribe data from the Edge Data Hub and process the data. For example, one can use our APIs to obtain the application data through the `application id` using a simple Python program as follows:

```
parser = argparse.ArgumentParser()
parser.add_argument('--app_id',help='application id')
args = parser.parse_args()
def on_edge_data(measurement):
    print("Got :")
    print(measurement)
iotrace.edge_dataproc_init()
mymainfunction = on_edge_data
iotrace.edge_data_processor(args.app_id,mymainfunction)
```

From the application id, `app_id`, our APIs connect to our middleware and subscribe the right Data Hub and the right Data Topic for the application.

Other typical application programming techniques can also be used. For example, Figure 4 shows an example of a Node-Red flow deployed in an edge server to read LoRaWAN data. After processing data, it pushes the result to the cloud. Given the DataTopic published by the Extractor, one can easily obtain the data from the topic and write some analytics. In the example, we filtered only alarms with id=312.

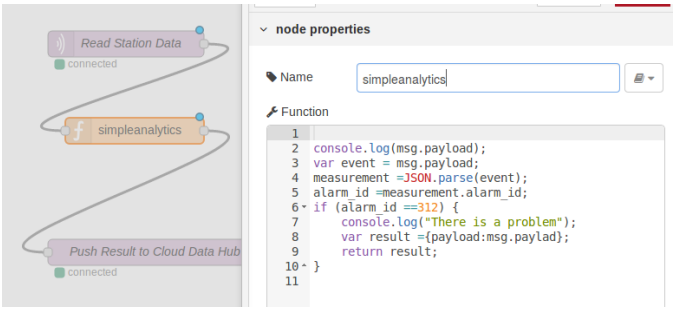

Fig. 4. Example of Node-RED -based analytics application

3) Cross Edge-Cloud Analytics: From the above-mentioned example, if the edge analytics pushes the results into other Data Hubs in the cloud, other application pipelines could utilize the results. Both Data Hubs in the edge and in the

cloud might utilizing the same technology, e.g., based on MQTT, simplifying the cross edge-cloud configuration. In this example, we do not have an end-user tool to show edge-cloud analytics so we demonstrate this capability by using existing tools. One can use Node-RED for edge analytics using an Edge Data Hub and the edge analytics can be done within any light weight computing node at the edge. After pushing the results into the Hub in the cloud, the following example shows an Apache Apex program that performs analytics of results from the edge.

```
//....
String topic ="application/test/iotrace";
MqttClientConfig btsmqttConfig = new MqttClientConfig();
//.... set configuration with the cloud data hub
StationMQTTInput btsInput = dag.addOperator("input",
     StationMQTTInput.class);
btsInput.setMqttClientConfig(btsmqttConfig);
btsInput.addSubscribeTopic(topic, qos);
ResultAggregrationOperator cons = dag.addOperator("
     resultaggregator", new ResultAggregrationOperator());
dag.addStream("test", btsInput.out, cons.input).setLocality
     (Locality.CONTAINER_LOCAL);
```

Although the examples show programming features using message queues, with application data available in Data Hubs in the edge, the application developer can utilize typical programming frameworks and APIs to access and analyze data from LoRaWAN in the edge. Other techniques, such as application deployment and programming frameworks, will be based on state-of-the-art data analytics programming.

VI. Related work

There are many open sources for LoRaWAN Gateways and Network Servers [9], [10]. They can be provided in the cloud or not. A few platforms have been introduced to support cloud-based connectivity for LoRaWAN. Main platforms are IoT-X [11] , LORIOT [12], and OrbiWise [13]. Usually a LoRaWAN gateway can be easy configured to work with different cloud connectivity platforms (as an example, Libelium allows different connectivity configurations with its Waspmote http://www.libelium.com/downloads/documentation/waspmote-lorawan-networking-guide.pdf). The Things Network (TTN) [2] is an open source community to build various middleware and services for IoT, especially LoRaWAN. In general, from the architecture and core services view, most of systems have similar concepts and services. However, they do not focus on supporting edge analytics. Our goal is to combine our work with existing platforms to enable edge analytics. Depart from such centralized data aggregation and analytics, first, we enable hybrid architecture for both edge and cloud applications by supporting configurable Network Servers pushing data to suitable edge and cloud Data Hubs. Second, we support diverse types of providers and data subscribers by concentrating on tool pipelines exploiting core features to deliver data to different applications.

Several examples of edge analytics have been presented but not with LoRaWAN [8], [4]. Mostly they deal with powerful networks and they are not designed to work with particular networks, mainly work on distributed data sources levels. We have designed our IoTRACE for LoRaWAN architecture, although our augmenting features could be adopted for other similar types of network architectures.

Existing frameworks have been proposed for edge analytics and for combining edge and cloud analytics, such as [14]. We focus on the middleware layer to enable data sharing and availability at the edge. Thus these frameworks can utilize our work to support different analytics applications.

VII. Conclusions and future work

We outlined IoTRACE for augmenting current LoRaWAN platforms to support edge analytics. In our work, we have augmented interactions among key components in LoRaWAN architectures and introduced new software components to support edge analytics. One of advantages of our approach is, by enabling edge data analytics of LoRaWAN, to bring LoRaWAN specific data to the same level with other types of data at the edge (e.g., from 4G and LAN); thus helping solving interoperability for data analytics at the edge. In fact, our proposal can also be investigated for other similar network architectures, such as SigFox. Nevertheless, our current paper is focused very much on software design perspective enabling edge analytics so it requires a substantial change in the architecture w.r.t. software components in (LoRWAN) gateways and network servers. It might require more powerful machines for acting as servers at the edge. Such a trade-off will be evaluated in the future work where we concentrate on our prototype.

Acknowledgment: This work was partially supported by the European Commission via the H2020 Inter-IoT project.

References

[1] "LoRaWAN, https://www.lora-alliance.org."
[2] "The Things Network, https://www.thethingsnetwork.org/," last access: 19 July 2017.
[3] T. Cao, H. Hoang, H. X. Huynh, B. Nguyen, T. V. Pham, M. Tran, V. T. Tran, and H. L. Truong, "Iot services for solving critical problems in vietnam: A research landscape and directions," *IEEE Internet Computing*, vol. 20, no. 5, pp. 76–81, 2016.
[4] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, Sep. 2015.
[5] E. Ahmed, A. Ahmed, I. Yaqoob, J. Shuja, A. Gani, M. Imran, and M. Shoaib, "Bringing computation closer toward the user network: Is edge computing the solution?" *IEEE Communications Magazine*, vol. 55, no. 11, pp. 138–144, NOVEMBER 2017.
[6] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos, "Edge analytics in the internet of things," *IEEE Pervasive Computing*, vol. 14, no. 2, pp. 24–31, Apr 2015.
[7] E. Ahmed, I. Yaqoob, I. A. T. Hashem, I. Khan, A. I. A. Ahmed, M. Imran, and A. V. Vasilakos, "The role of big data analytics in internet of things," *Comput. Netw.*, vol. 129, no. P2, pp. 459–471, Dec. 2017.
[8] Y. Cao, P. Hou, D. Brown, J. Wang, and S. Chen, "Distributed analytics and edge intelligence: Pervasive health monitoring at the era of fog computing," in *Proceedings of the 2015 Workshop on Mobile Big Data*, ser. Mobidata '15. New York, NY, USA: ACM, 2015, pp. 43–48.
[9] "LoRa Server, https://github.com/brocaar/loraserver."
[10] "LoRaWAN Server, https://github.com/gotthardp/lorawan-server."
[11] "IoT-X Connectivity Platform, https://www.stream-technologies.com/iotx/," last access: 19 July 2017.
[12] "LORIoT, https://www.loriot.io/," last access: 19 July 2017.
[13] "Orbiwise, https://www.orbiwise.com," 19 July 2017.
[14] B. Cheng, A. Papageorgiou, F. Cirillo, and E. Kovacs, "Geelytics: Geo-distributed edge analytics for large scale iot systems based on dynamic topology," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, Dec 2015, pp. 565–570.