MobIoTSim: Towards a Mobile IoT Device Simulator

T. Pflanzner, A. Kertesz
Software Engineering Dept., University of Szeged
H-6720 Szeged, Dugonics ter 13, Hungary
Email: {tampfla,keratt}@inf.u-szeged.hu

B. Spinnewyn and S. Latré
University of Antwerp – iMinds
2020 Antwerp, Middelheimlaan 1, Belgium
Email: {bart.spinnewyn,steven.latre}@uantwerpen.be

***Abstract*—Currently a growing number of powerful devices join the Internet composing a world of smart devices, or things in the Internet of Things (IoT) perspective, significantly impacting the global traffic. There are also more and more cloud providers offering IoT-specific services, since cloud computing has the potential to satisfy IoT needs such as hiding data generation, processing and visualization tasks. Using the capabilities of smartphones, many things can be simulated simultaneously supporting most types of IoT devices. In this paper, we discuss the design and development of a mobile IoT simulator called MobIoTSim that helps researchers to learn IoT device handling without buying real sensors, and to test and demonstrate IoT applications utilizing multiple devices. We also show how to develop a gateway service in a cloud that can be connected to MobIoTSim to manage the simulated devices and to send back notifications by responding to critical sensor values. By using this tool, developers can examine the behaviour of IoT systems, and develop and evaluate IoT cloud applications more efficiently.**

1. Introduction

The Cluster of European Research Projects on the Internet of Things [5] considers the Internet of Things (IoT) as a vital part of Future Internet and they defined it as a dynamic global network infrastructure with self configuring capabilities based on standard and interoperable communication protocols. Things in this network interact and communicate among themselves and with the environment by exchanging sensor data, and react autonomously to events and influence them by triggering actions with or without direct human intervention. According to another Gartner report [3] there will be 30 billion devices always online and more than 200 billion devices discontinuously online by 2020.

These trends and estimations call for an ecosystem that provides means to interconnect and control these devices. With the help of cloud solutions, user data can be stored in a remote location, and can be accessed from anywhere. The concept of cloud computing has been pioneered by commercial companies with the promise to allow elastic construction of virtual infrastructures, which attracted users early on. Its technical motivation has been introduced in [1]. Gubbi et al. [4] have identified that to support the IoT vision, the current computing paradigm needs to go beyond traditional mobile computing scenarios and cloud computing has the potential to address these needs as it is able to hide data generation, processing and visualization tasks. For this reason, there are more and more PaaS cloud providers offering IoT specific services (e.g. Google Cloud Platform and IBM Bluemix Platform). Some of these IoT features are unique, but every IoT PaaS provider has the basic capabilities to connect and store data from devices. Many things have to be managed at the same time and a wide range of devices and data formats are available, therefore creating and examining such applications are not trivial.

The aim of our research is to support the proliferation of IoT with the help of mobile and cloud technologies, thus to enable experimenting with complex systems consisting of interdependent components that work together to enable the creation and management of user applications. To manage the heterogeneity of protocols and data structures used in these IoT cloud systems, smartphones and tablets can provide assistance. In this paper, we discuss the design and development of a mobile IoT simulator called MobIoTSim that is capable of managing many things using the most popular IoT protocols (such as MQTT, HTTP) combined with different data structures/formats (e.g JSON). It can substitute real things or devices by mimicking their behaviour. Therefore the purpose of our proposed mobile IoT device simulator, is to help researchers and cloud application developers to learn IoT device handling without buying real sensors, and to test and demonstrate IoT applications utilizing multiple devices.

We also introduce how to develop a cloud gateway service to manage the simulated devices and to send notifications to MobIoTSim by responding to critical sensor values. Cloud federations can also be applied to serve the needs of IoT applications, as discussed by Celesti et al. [6]. By deploying such gateway services to different providers, we can envision an inter-cloud IoT scenario, in which the simulated devices can communicate to different clouds (e.g. due to privacy or jurisdictional restrictions).

Further advantages of this mobile approach are that mobile devices can act as micro-gateways to capture IoT data

from sensors representing a semi-simulated environment, and outdoor IoT deployments can also be designed and evaluated in a more realistic way (e.g. using wireless mobile data communication instead of WiFi).

The remainder of this paper is structured as follows: Section 2 presents a survey of related works, and Section 3 presents the vision and requirements of our proposed approach to design a mobile IoT device simulator. Section 4 introduces our solution, demonstrates its utilization with cloud gateways, and highlights future development plans. Finally, the contributions are summarized in Section 5.

2. Related work

The increasing competition between the leading vendors in the cloud market, such as Amazon, Microsoft, Google and SalesForce, each of which promotes its own, mostly incompatible cloud standards and formats [7], prevents them from agreeing on a widely accepted, standardized way to utilize cloud details and specifications, which trend also appeared in the IoT field. However, an interoperable cloud environment would benefit customers, as they could migrate their virtual machines, data and applications between cloud providers without setting data at risk.

In the field of resource abstraction for IoT, good efforts have been made towards the description and implementation of languages and frameworks for efficient representation, annotation and processing of sensed data. The integration of IoT and clouds has been envisioned by Botta et al. [11] by summarizing their main properties, features, underlying technologies, and open issues. A solution for merging IoT and clouds is proposed by Nastic et al. [12]. They argue that system designers and operations managers face numerous challenges to realize IoT cloud systems in practice, due to the complexity and diversity of their requirements in terms of IoT resources consumption, customization and runtime governance. These related works also serve as a motivation to our research by raising the need for managing a large number of protocols and data formats by means of simulation.

The existing simulators used to examine IoT systems are general network simulators, e.g. NetSim [8], Qualnet [9] and OMNeT++ [10]. With these tools IoT-related processes can be examined such as device placement planning and network interference. The OMNeT++ discrete event simulation environment [10] is a generic tool for simulating communication networks, multiprocessors and other distributed systems. It can be used in numerous domains from queuing network simulations to wireless and ad-hoc network simulations, from business process simulation to peer-to-peer networks.

There are some more specific IoT simulators closer to our approach. Han et al. have designed DPWSim [15], which is a simulation toolkit to support the development of service-oriented and event-driven IoT applications with the aim to support the OASIS standard Devices Profile for Web Services (DPWS). Though this enables the use of web services on smart and resource-constrained devices, it also limits its application scope. The SimpleIoTSimulator [13] is an IoT Sensor/device simulator that is able to create test environments made up of thousands of sensors and gateways on a computer. It supports many of the common IoT protocols (e.g. CoAP, MQTT, HTTP). Its drawback is that it needs a 64-bit RedHat Enterprise Linux environment to be installed. Our approach is more focused on IoT device simulation with smartphones, which is easier to be combined with real world applications. The Atomiton simulator [14] seems to be the closest to our concept. It simulates virtual sensors, actuators and devices with unique behaviours. With this tool complex, dynamic systems can be created for specific applications. Unlike our open mobile solution, it is a commercial, web-based environment with very limited documentation.

3. The need for an Internet of Things simulator

The motivation behind our research is that more and more PaaS cloud providers have started to offer IoT specific services to ease the development of IoT cloud applications, but cases where many heterogeneous things need to be managed are hard to realize and examine. For example, smart city application scenarios using LoRa [16] or SIGFOX [17] technologies are very expensive and time-consuming to set up with real devices (a basestation costs more than a thousand Euros with a lot of configuration work). Therefore we propose to use simulated devices with a cloud gateway in order to ease the development and testing of such systems. The next subsection introduces and compares current PaaS cloud IoT capabilities, and the following details requirements for such a simulator.

3.1. PaaS cloud providers with IoT support

The IBM Bluemix Platform [18] is an IoT-enabled PaaS solution offered by IBM. It can be used for quick development of cloud-based applications that take advantage of the data generated by the sensors and devices. Products of several major device manufacturers are supported, such as ARM, the Electronics B&B, Intel, Multi-Tech Systems and Texas Instruments, but other individual cases can also be solved on the platform. Data generated by the equipment is sent by the popular and lightweight MQTT protocol to the cloud. The service allows the users to configure, manage the devices, and to store the history of generated data or stream real-time data to the application. The data transfer can be done through secure APIs.

To illustrate the inner workings of the platform, a real-time data visualization demo is also provided. To use it, first a data provider should be configured, which is in the simplest case a smartphone, but it is possible to use a TI SensorTag, ARM Mbed, Raspberry Pi, Intel and other devices. The opened browser page on the smartphone can send real-time data of the phone's movement to the cloud application. The framework also provides a pre-defined web-based sensor simulator [19] that is able to act as three simulated sensors (sending temperature and humidity values through websockets).

TABLE 1. PAAS IOT FEATURES.

Provider	Open source	Hosting	SDK	Data store	BLOB	GEO	Push not.	Trigger	Visualization
Bluemix	no	closed	JS	yes	no	yes	yes	yes	yes
Parse	yes - SDK	closed	C	yes	yes	yes	yes	yes	no
Google	no	closed	JS	yes	N/A	N/A	yes	yes	yes

The Bluemix platform offers several specialized services to support the development of cloud applications. Some examples of these services are: Push for messaging, Cloudant NoSQL DB to manage NoSQL databases, Geospatial Analytics for location tracking, and IBM Analytics for Hadoop for Hadoop computations.

The supported languages for application development are Java, JavaScript, GO, PHP, Python and Ruby. In terms of costs, a price calculator helps to determine a monthly fee for a 30-day trial period. 20 devices can be connected and 100MB of data can be sent to the devices for free, which is enough for about 50,000 messages. 1GB storage space can also be used in this period.

The Parse [21] PaaS platform developed by Facebook also has IoT support. This platform promises quick and easy application development with the support for mobile devices (through its MBaaS service). In addition to the C SDK for Linux (Raspberry Pi) and real time systems (RTOS) (TI CC3200), there are some more special versions offered such as Arduino, and a number of partners have SDKs, such as Atmel, Broadcom, Intel and Texas Instruments. A great benefit of the platform is that all of their SDKs are open source. These SDKs allow data sending and the so-called push notifications, and they can also take advantage of the Parse cloud services. Many sample applications are available to demonstrate its usage, including farming, music and cooking scenarios. JavaScript applications are supported, and the Parse Webhook service makes it able to link applications from remote clouds. The supported mobile platforms are the iOS, Windows Phone, Android, Unity and Xamarin. Web and desktop SDKs are for OSX, Windows, JavaScript, Unity, PHP, and .NET. The SDK supports offline data storing, social media connections can be properly handled. It is possible to schedule jobs, as well as a high quality Dashboard is also available that supports data modification, statistical testing, push notification, and storing logs. Objects can be managed in the system after specifying their schema. Their schema supports many field formats, e.g. arbitrary objects, array, blobs and geopoints. Their dashboard provides no tools for data visualization. Blobs can be imported from a public URL, even directly from the Base64 encoded Data URI format.

The Google IoT solution [20] is part of the Google Cloud Platform, which includes various Google services. The scalability is an excellent feature in this platform. It allows devices to be connected, and it collects data and visualizes them. The data sent from the devices are received by the Google Load Balancer and forwards to instances of the AppEngine applications. In general, the main part of the application is the AppEngine, which may use other services. Compute intensive tasks are supported by the Compute Engine. The Cloud Storage and the Cloud SQL manages data. It is possible to send data with streams to the BigQuery service, which is ideal if we want to work with real-time data. In IoT systems the visualization is an important feature, it is supported in real time using the Google Charts. Google is also strong in managing a large amount of data processing, which is important, since there are many devices generating huge amount of data in IoT systems. Google FireBase plays an important role in the management of the devices. It was originally designed to assist mobile devices (like MBaaS). It provides synchronized real-time database, authentication and capable of offline operations.

The main IoT-related properties of these cloud providers are summarized in Table 1. Bluemix has the highest variety of IoT-related services. Parse has its high quality MBaaS service, while its IoT feature is a limited version of it. IoT applications in Google can be composed of many connected services, which makes it complex providing more freedom for the developers. They are also very good at scaling and performance, and this complexity is compensated by the simplicity of FireBase.

From this survey we can see that the most popular cloud providers have already realized the need for IoT support, and most of them provide reasonably good solutions for IoT application development. Nevertheless interoperability issues still exist, and applications managing a large number of different IoT devices are hard to develop and evaluate. Bluemix has also identified the need for a sensor or device simulator, but its tool is meant to serve simple demonstration purposes. Our aim is to design a generic solution to this problem.

3.2. Requirements for a cloud-enabled simulator

We identified the following incremental challenges related to IoT networks:

- IoT devices are battery powered;
- They communicate using low-power wireless technologies (e.g., IEEE 802.11, IEEE 802.15.4, Bluetooth);
- There are different resource constraints of devices (e.g., on CPU, memory, connectivity);
- IoT networks are very dynamic as network conditions can change rapidly;
- They are heterogeneous: there is a large spread on device capabilities (e.g., powerful cameras, low cost temperature sensors), additionally there are sources (sensors) and sinks of information (actuators);

- They are very dynamic: the networking environment in a IoT environment is largely unstructured can vary rapidly.

There are different kind of IoT environments, hence their static or dynamic properties, and the number of utilized devices can affect the design of such a simulator. For example, a connected house can be regarded as a static environment, because its devices are usually in one place, possibly with wired connection, providing reliable network stability. The dynamic environment is more complex to simulate, in such cases we would like to simulate a broader part of the environment considering WiFi interference, battery lifetime and locations of the devices.

We are not aiming at simulating whole IoT systems and networks, but we still want to aid the design, development and testing processes of these systems. Our goal is to develop a mobile IoT device simulator that can emulate real devices and sensors, thus it can be used in the previously mentioned processes instead of real resources.

The requirements for basic functionalities of such a simulator are to send and receive messages, generate sensor data (for one or more devices), and react to received messages. These capabilities are sufficient to use the simulator in IoT system analysis. Requirements for advanced functionalities such as simulating network errors, recording and replaying concrete simulation cases, and connecting real IoT devices to the simulator can contribute to the analysis of more realistic system.

In our work we planned to support the basic functionalities with the following settings:

- a simulated device should have an ID, or tokens for authentication;
- the generated sensor data should be made available in binary, plain text, or JSON format with metadata like date, time, and device state;
- finally MQTT or REST communication protocols should be supported.

4. The Mobile IoT Device Simulator

The main purpose of MobIoTSim, our proposed mobile IoT device simulator, is to help cloud application developers to learn IoT device handling without buying real sensors, and to test and demonstrate IoT applications utilizing multiple devices. The structure of the application lets users create IoT environment simulations in a fast and efficient way with the options for custom settings.

4.1. Architecture

Our mobile IoT device simulator can simulate one or more IoT devices, and it is implemented as a mobile application for the Android platform. Sensor data generation of the simulated devices are random generated values in the range given by the user. The data sending frequency can also be specified for every device. The application uses MQTT protocol to send the data with the use of the Eclipse Paho opensource library. The data is represented in a structured JSON object compatible with the IBM IoT Foundation message format [22].

Screenshots of the simulator can be seen in Figure 1. After starting the MobIoTSim Android application, the user can navigate to the Cloud settings or the Devices screens. The first one can be used to define the connection parameters of a gateway residing in a cloud. These parameters are: organization ID, URL, port and connection type for communicating with its MQTT server, and optional parameters: application ID, auth key and auth token e.g., for accessing visualized data. Users can also select predefined settings from templates, an example parameter setting for the Bluemix Quickstart demo gateway can be seen in Figure 1.a.

The Devices screen shows the list of currently simulated devices. Each device can be started/stopped or edited (see Figure 1.b). A new device can be added by clicking on a button below the list. The creation of a new device and editing the details of an existing device is managed by the same screen, the Device settings screen (see Figure 1.c). Here the user can specify a device type, an ID of the device, a token (to authenticate with the MQTT server of the gateway), the data generation frequency, and the range for random numeric value generation.

4.2. Evaluation

In order to exemplify the usability of MobIoTSim, we connected it to the Quickstart application (i.e. demo gateway) of the IBM IoT Foundation with an MQTT server [23] (with the settings shown in Figure 1.a). Once we registered a simulated device to the MQTT server of the IBM IoT Foundation system, and started it in MobIoTSim (like MobIoT_test01 as shown in Figure 1.b), the data generated by the device is continuously sent to the demo gateway. A screenshot of the received and visualized data in the IBM IoT Foundation demo gateway can be seen in Figure 2.

We also developed an own gateway service in the IBM Bluemix Platform that is able to manage more devices simultaneously, and to send a notification to the MobIoTSim device simulator by responding to critical sensor values. This gateway service is basically an extended version of the IOT Visualization application [24] of the IBM Internet of Things Cloud. It has a web-based graphical interface to visualize sensor data coming from MobIoTSim. Messages (defined in JSON format) received from the simulated devices are managed by an MQTT server. It can also be used to send responses (or notifications) back to the simulated IoT devices in MobIoTSim.

Figure 3.a shows how to connect the simulator to this gateway. Since it has a predefined template called Bluemix, we only need to specify an organization ID and the connection type (TCP or secure TLS) (the URL is given in the template) to enable connection to the MQTT server, while the application ID, the auth key and token can be retrieved by registering to the gateway service (these parameters can

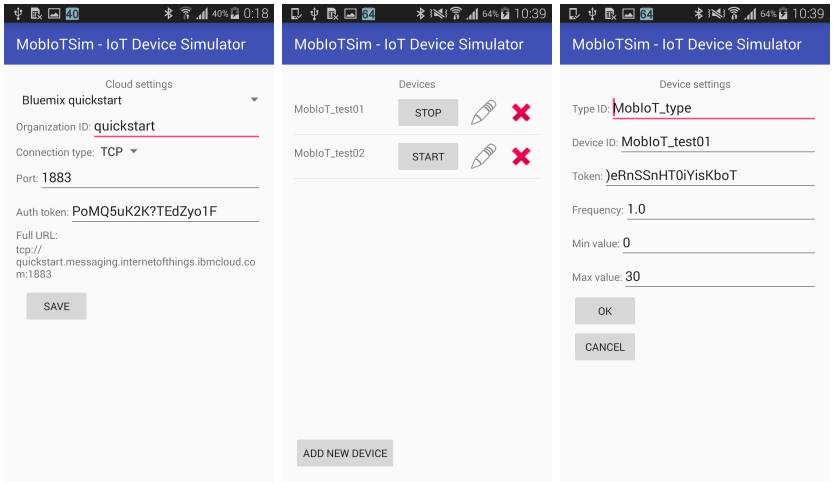

Figure 1. Screenshots of the MobIoTSim Android application: a.) Cloud settings, b.) Devices, c.) Device settings

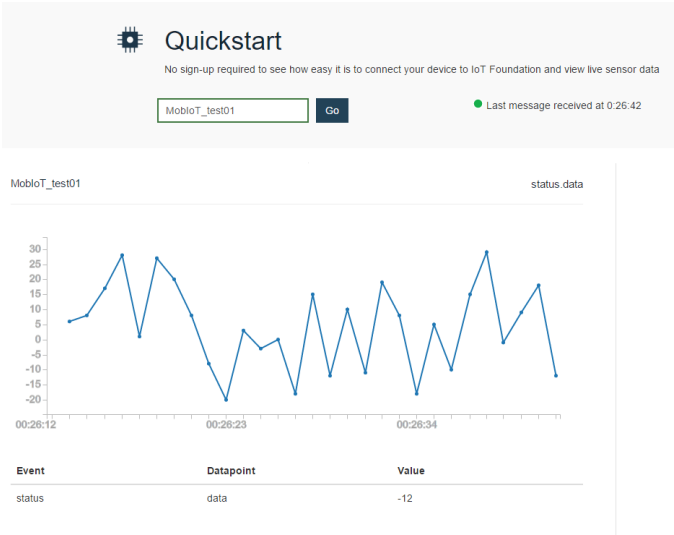

Figure 2. Data visualization in the IBM IoT Foundation demo gateway.

be used later to sign in to the data visualization site of the gateway). The simulated devices also need to be registered to the MQTT server of the gateway service by specifying their device and type identifiers and sensor data thresholds, which replies with their token identifiers (to be used for device setting as shown in Figure 1.c). Once these settings are made, simulated devices can be defined and started in the same way as shown previously for the demo gateway. With this own gateway we can create advanced scenarios, such as managing more devices and responding to critical sensor data coming from the simulated devices. Figure 3.b shows a situation in which a warning message is sent to a device (named MobIoT_test01 in MobIoTSim), when sensor data values are over/under a predefined threshold. Figure 4 shows the GUI of the own Bluemix gateway service by depicting the data received from a selected simulated device.

4.3. Future Extensions

We plan to extend MobIoTSim in several directions. First, we believe that gateway templates could provide useful means for experimenting with the simulator. Currently the previously introduced gateway services for the IBM Bluemix Platform and Azure IoT Hub are available, but we plan to support additional, popular cloud providers, e.g. Amazon and Google. A general, platform independent gateway would also be useful. It could be realized with a cloud visualisation application consisting of three main parts: the data collection, the database and the visualisation part. We already started to design such a general solution, where data collection is managed by an MQTT broker and a REST server. The database could be excluded, if the data storage is not necessary and the data streaming is possible directly to the visualisation part, but for advanced features like displaying data history or statistical reports it

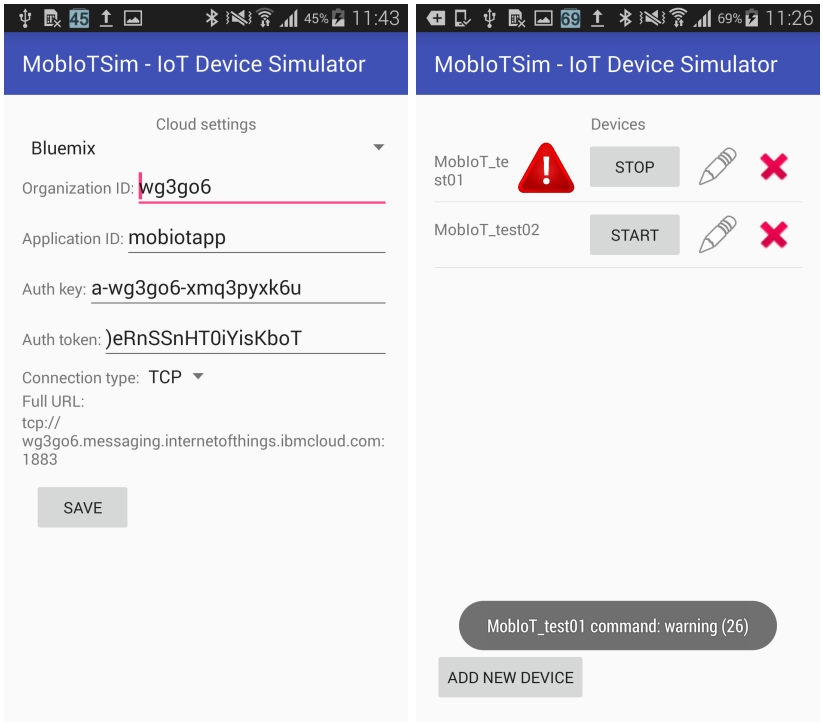

Figure 3. Screenshots for using an own gateway: a.) Cloud settings, b.) Devices screen showing a warning

is useful. The visualisation part will be supported by live charts showing the data coming from the IoT devices in real time, and statistical data will also be available.

We also plan to support larger, scalable experiments in the future. For this purpose we will use a scripting language to be able to specify the devices and their properties, and to schedule their activities. For cloud application testing, a great feature would be to record sensor data and networking events, and later it could be replayed again many times, with exactly the same scenario.

The network errors are common problems in real IoT systems, therefore we also plan to investigate this issue in more detail. The current version of MobIoTSim can only generate random errors upon request (with a static setting). However, more complex simulations can also model the IoT network more accurately by considering wireless interference and propagation models. Those networking conditions are typically considered by wireless network simulators such as NS-3 [25]. While such an extension makes the simulations more realistic, it will also significantly increase setup time and computation time of such simulations.

5. Conclusion

In this paper we presented our results towards developing a general purpose IoT device simulator. We introduced the requirements and performed design steps of our mobile IoT simulator called MobIoTSim that is capable of simulating more IoT devices by generating real time sensor data. We have also developed an own gateway service in the IBM Bluemix Platform that can be connected to MobIoTSim to manage the simulated devices and to send notifications to the simulator by responding to critical sensor values.

By using this tool, researchers and developers can examine the behaviour of IoT systems, and develop and evaluate IoT cloud applications more efficiently. Our future work will address the extension of MobIoTSim with pre-defined data generation templates, and a basic propagation and loss model to simulate the network transmission. Further gateway developments for different cloud providers are also planned to ease integration with other clouds and foster inter-cloud deployments.

Acknowledgments

The research leading to these results has received funding from the European COST programme under Action identifier IC1304 (ACROSS), and it was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

References

[1] Buyya B, Yeo C S, Venugopal S, Broberg J, and Brandic I, Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Generation Computer Systems, vol. 25, no. 6, pp. 599-616, June 2009.

[2] Pring B et. al., Forecast: Public Cloud Services, Worldwide and Regions, Industry Sectors, 2009-2014. Gartner report. Online: http://www.gartner.com/Display-Document?ref=clientFriendly-Url&id=1378513, June 2010.

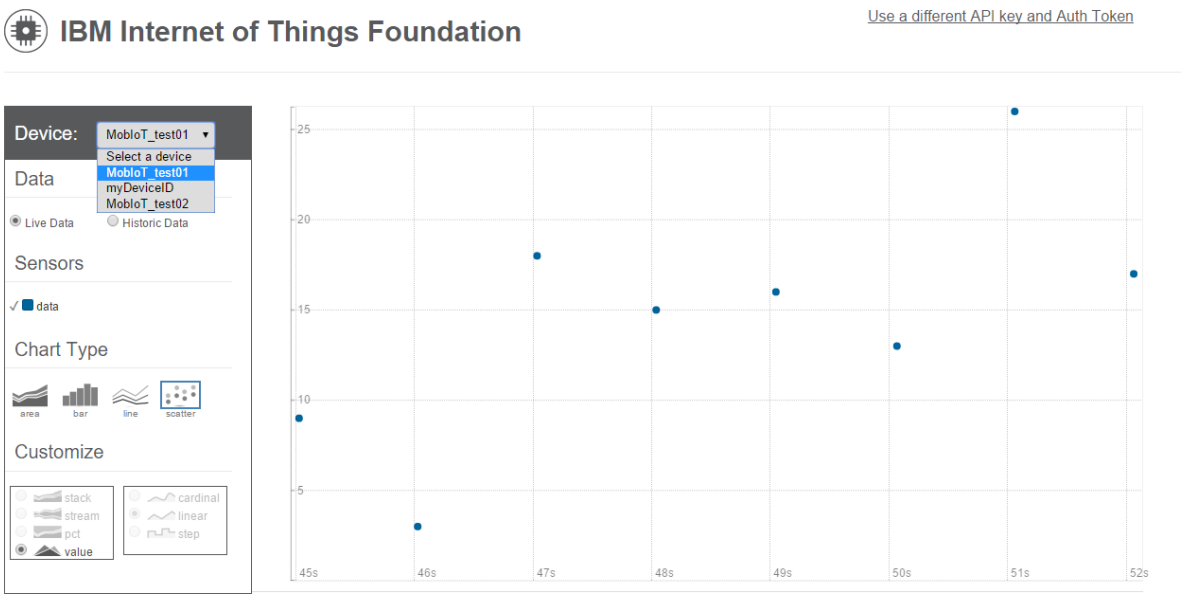

Figure 4. Data visualization in an own gateway service in Bluemix.

[3] J. Mahoney and H. LeHong, The Internet of Things is coming, Gartner report. Online: https://www.gartner.com/doc/1799626/internet-things-coming, September 2011.

[4] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, Volume 29, Issue 7, pp. 1645-1660, September 2013.

[5] H. Sundmaeker, P. Guillemin, P. Friess, S. Woelffle. Vision and challenges for realising the Internet of Things. CERP IoT - Cluster of European Research Projects on the Internet of Things, CN: KK-31-10-323-EN-C, March 2010.

[6] A. Celesti, M. Fazio, M. Giacobbe, A. Puliafito, and M. Villari, Characterizing Cloud Federation in IoT. IEEE 30th International Conference on Advanced Information Networking and Applications Workshops - Workshop on Cloud Computing Project and Initiatives, 2016.

[7] G. Sperb Machado, D. Hausheer, and B. Stiller, Considerations on the Interoperability of and between Cloud Computing Standards. In 27th Open Grid Forum (OGF27), G2CNet Workshop: From Grid to Cloud Networks, Banff, Canada, 2009.

[8] Boson NetSim Network Simulator. Online: http://www.boson.com/netsim-cisco-network-simulator. Accessed at January, 2016.

[9] QualNet communications simulation platform. Online: http://web.scalable-networks.com/content/qualnet. Accessed at January, 2016.

[10] A. Varga and R. Hornig, An overview of the OMNeT++ simulation environment. In Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops (Simutools '08). 2008.

[11] A. Botta, W. de Donato, V. Persico, A. Pescape, On the Integration of Cloud Computing and Internet of Things. The 2nd International Conference on Future Internet of Things and Cloud (FiCloud-2014), August 2014.

[12] S. Nastic, S. Sehic, D. Le, H. Truong, and S. Dustdar, Provisioning Software-defined IoT Cloud Systems. The 2nd International Conference on Future Internet of Things and Cloud (FiCloud-2014), August 2014.

[13] SimpleSoft SimpleIoTSimulator. Online: http://www.smplsft.com/SimpleIoTSimulator.html. Accessed at January, 2016.

[14] Atomiton IoT Simulator. Online: http://atomiton.com/simulator.html. Accessed at January, 2016.

[15] S. N. Han, G. M. Lee, N. Crespi, N. V. Luong, K. Heo, M. Brut, P. Gatellier, DPWSim: A simulation toolkit for IoT applications using devices profile for web services. In proc. of IEEE World Forum on Internet of Things (WF-IoT), pp.544-547, 6-8 March 2014.

[16] LoRa Technology. Online: https://www.lora-alliance.org/What-Is-LoRa/Technology. Accessed at January, 2016.

[17] SIGFOX. Online: http://www.sigfox.com/en/#!/connected-world. Accessed at January, 2016.

[18] IBM Bluemix Platform. Online: https://console.ng.bluemix.net/. Accessed at January, 2016.

[19] IBM Bluemix IoT Sensor. Online: https://developer.ibm.com/recipes/tutorials/use-the-simulated-device-to-experience-the-iot-foundation/. Accessed at March, 2016.

[20] Google Cloud Platform. Online: https://cloud.google.com/solutions/iot/. Accessed at January, 2016.

[21] Parse. Online: https://parse.com/products/iot. Accessed at March, 2016.

[22] IBM IoT Foundation message format. Online: https://docs.internetofthings.ibmcloud.com/gateways/mqtt.html#/managed-gateways#managed-gateways. Accessed at January, 2016.

[23] IBM IoT Foundation Quickstart application. Online: https://quickstart.internetofthings.ibmcloud.com. Accessed at January, 2016.

[24] IOT Visualization application. Online: https://github.com/ibm-messaging/iot-visualization/. Accessed at January, 2016.

[25] NS-3. Online: https://www.nsnam.org/. Accessed at March, 2016.