Internet of Things data management in the cloud for Bluetooth Low Energy (BLE) devices

Theodoros Soultanopoulos
Department of Electronic and Computer Engineering
Technical University of Crete
73100, Greece
(+30) 28210 37229
tsoultanopoulos@gmail.com

Stelios Sotiriadis
Electrical and Computer Engineering, Computer Engineering Research Group, University of Toronto
St. George Campus, 40 St George St, Toronto, ON M5S 2E4, Canada
(+1) 4168285148
s.sotiriadis@utoronto.ca

Euripides Petrakis
Department of Electronic and Computer Engineering
Technical University of Crete
73100, Greece
(+30) 28210 37229
petrakis@intelligence.tuc.gr

Cristiana Amza
Electrical and Computer Engineering, Computer Engineering Research Group, University of Toronto
St. George Campus, 40 St George St, Toronto, ON M5S 2E4, Canada
(+1) 4168285148
amza@ece.utoronto.ca

ABSTRACT

The use of wearable sensors and their connectivity to Internet offers significant benefits for storing sensing data that could be utilized intelligently for multiple purpose applications such as for monitoring purposes in healthcare domain. This work presents an Internet of Things (IoT) gateway service taking advantage of modern mobile devices and their capabilities to communicate with wearable Bluetooth low energy (BLE) sensors so data could be forwarded to the cloud on the fly and on real time. The service transforms a mobile platform (such as a smartphone) to a gateway allowing continuous and fast communication of data that is forwarded from the device to the cloud on demand or automatically for an automated decision making. Its features include (a) use of an internal processing mechanism for the BLE sensor signals and defines the way in which data is send to the cloud, (b) dynamic service as it has the ability to recognize new BLE sensors properties by easily adapting the data model according to a dynamic schema and (c) universal BLE devices capability that are registered automatically and are monitored on the fly while it keeps historical data that could be integrated into meaningful business intelligence. Building upon principles of service oriented design, the service takes full advantage of cloud services for processing potential big data streams produced by an ever increasing number of users and sensors. The contribution of this work is on the IoT data transmission rate that is averagely calculated to 128 milliseconds and in the experimental section we discuss that this is significantly low for real time data.

CCS Concepts

• Software system structures →Distributed systems organizing principles →Cloud computing • Software and its engineering →Software organization and properties →Software systems architectures →Distributed systems organizing principles →Cloud computing • Hardware →Communication hardware, interfaces and storage →Sensor applications and deployments

Keywords

Cloud Computing, Internet of Things, Sensor data collection service, Bluetooth Low Energy (BLE), REST services.

1. INTRODUCTION

The Internet of Things (IoT), combined with the cloud computing opens new opportunities in application domains relying on the idea of real time monitoring of users' activities (e.g. sports activities), ambient living or health status monitoring for risk prevention and improved health care [2]. Cloud computing provides the means for scalable data storage and on the fly processing of monitored information [4]. This work exploits the opportunities offered by modern mobile devices "to be transformed" to a communication bus (gateway) between Bluetooth Low Energy (BLE)[1] portable sensors and cloud systems. We envisioned that BLE has a "high potential of becoming an important technology for the IoT in low power, low cost, small devices" [10]. This work proposes a cloud service architecture that allows collection, processing and storage of IoT data produced by BLE devices. The service involves a cloud based implementation for receiving, processing and storing non structured data in real time in NoSQL format (e.g. in JSON). It further combines general purpose services referred to Generic Enablers (GEs) [1]. The solution is realized as a service oriented architecture that gives significant benefits over traditional systems including scalability, elasticity and more effective management.

BLE protocol, compared to the traditional Bluetooth provides similar communication range at a considerable reduced power consumption cost [6], [10]. This work focuses on the problem of transforming BLE devices to "services" that could be easily linked to a mobile device for data pushing. The aim of this work is to "transform" content and context from BLE devices to a flexible and generic API that will be easily accessible without worrying for internal processes for data structures or schemas e.g. IoT non-structured or semi-structured data [13]. We are motivated by the work of [7] that presents challenges in discovery and control of IoT data. Also, the work of [14] suggests that "cloud platforms need to be enhanced to support the rapid creation of applications, by providing domain specific programming tools and environments and seamless execution of applications, harnessing

[1] https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth-technology-basics/low-energy

capabilities of multiple dynamic and heterogeneous resources". We design and develop an IoT gateway that could be installed in a mobile device and can connect to cloud systems, so to obtain different sensor signals and with appropriate mechanisms to send data to different services developed in the cloud. The service is tracking user sensing data in real time and by receiving and sending data produced by sensors based in an automation method. The contribution of this work is as follows.

(a) It supports universal usage over BLE sensors that could be attached easily to any mobile device (that has the service preinstalled on it) thus makes feasible to develop cloud applications by decoupling the system logic from the industry specific sensors.

(b) It allows automated data collection from all BLE devices following a simple configuration of an XML schema for registering sensors. Also, it supports historical data storage. The IoT data transmission rate is calculated to 128 milliseconds and is significantly low for real time data.

(c) It is user and sensor tailored and automated based on the challenge presented in [11] so data collected in an IoT system needs to be easily associated with a user, thus sensor driven decision analytics become easy and supports situational awareness.

Next Section 2 presents the background and the motivation of our work, Section 3 the service architecture, Section 4 the implementation and Section 5 the performance evaluation. Section 6 presents the conclusions and the future research steps.

2. Background

Cloud computing provides an environment where a large range of hardware and software services are built upon the concepts of scalability and elasticity. As such, it is the ideal environment for IoT applications design and implementation for accommodating the needs of an ever increasing number of users and devices along with their needs for management (i.e. storage, processing) of large streams of data acquired by these devices [8]. However, "data transmission is the dominant energy consuming operation in a sensor node" [9]. Smart applications developed for example in the concept of a smart city, could gain advantage of the flexibility and elasticity that cloud computing technology could offer in addition to the low energy consumption of BLE devices. The development of specific services in Cloud Computing environments for the management of IoT equipment is already in development. In this work we focused on the problem of sensor data collection in cloud systems in an automated and on the fly way, thus we propose a gateway.

The service architecture is based on the FIWARE platform[2] that is a non-commercial platform that offers general purpose services (called GEs) [1]. FIWARE introduced an innovative infrastructure for cost-effective creation and delivery of of Future Internet (FI) applications and services [1]. The functionality and specification of GEs provided by FIWARE can be accessed through a public catalogue[3], in a way that developers can easily browse and select the appropriate APIs to use. We utilize the BLE protocol that is a standard for low-power devices with low latency and short-range networks (<50m). Its characteristics are (a) low power consumption, (b) low cost and (c) balancing computing power[4]. The BLE Attribute Protocol describes the basic information structure, the information discovery process from a remote device and the ability to interact with the generated information. It provides a Generic Access Profile (GAP) that defines roles, e.g. which sensors that are compliant with the same standard can discover each other and exchange data. It also includes the Generic Attribute Profile (called as GAAT)[5] that defines the way in which devices that are following the BLE protocol can transfer data when are paired. It further uses the attributes provided by GAP, organizing them into a hierarchy of data structures. The format includes the "profile" (an abstract concept that includes services), that divide information into logical pieces e.g. heart rate service and the "features" that are information specific to a particular sensor (e.g. heart rate measurement).

3. Gateway Service Architecture

This section presents the gateway service architecture including (a) the IoT reference architecture, (b) the IoT gateway service and (c) its data flow analysis. We use the reference architecture to design a use case specific system for BLE sensors and we present the information flow among its components.

3.1 IoT Referece Architecture

The conceptual model of a service-centralized architecture may involve different cloud service providers that develop modules, where each follows its own development principles and tools (e.g. operating system, programming languages, natural resources, etc.) [5]. A reference architecture represents a model of groups of services that are divided over four main domains including the producers, the front-end, the back-end and the consumers. Producers comprise sensors that generate data (in our case we focus on devices embedded with BLE sensors). The front-end includes the gateway that plays the role of a mediator between the data sent by the sensor and the data managed by the service.

This data is forwarded to the back-end system that includes the general purpose services for user authentication, data context subscription. These are usually deployed as modular cloud services that may belong to heterogeneous cloud providers. All these services are developed around the application logic, which makes use of standards, controls and conditions for the transfer of information on individual services and orchestrates the so called business intelligence of the service. Finally, consumers are either end users or other applications.

3.2 Gateway Service Architecture

The implementation of the IoT gateway service it is based on the service oriented architecture of Figure 1. The architecture is modular comprising of individual services as described in Section 3.1. In our design the "Sensor Data Collector" module as a mobile service for BLE devices including the next. The producers are users that have BLE sensors that produce information about their vital data e.g., blood oxygen saturation and pulse rate in real time). In the case of BLE sensors so whenever a new device is activated, it starts the process connection to a host device and data subscription.

[2] https://www.fiware.org

[3] http://catalogue.fiware.org/

[4] https://developer.bluetooth.org/TechnologyOverview

[5] https://developer.bluetooth.org/TechnologyOverview/Pages/GATT.aspx

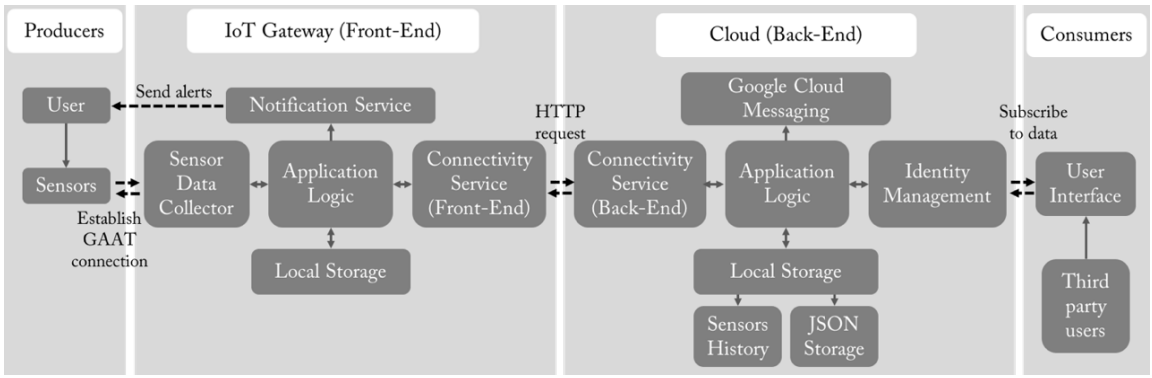

Figure 1: Architecture of the BLE IoT Service data collection and processing

The gateway service encapsulates the properties of a middleware via a mobile device to transfer the data from the sensors to the cloud. Users can interact with the gateway to setup data threshold rules. The cloud back end system is the place to host the data obtained from the front end. Data is processed and stored appropriately based on the use case scenario in order to serve the user needs. The consumers are the users who require to have access to data produced by the system and the application interface. This includes graphical user interfaces that facilitates third party users access to information and services. Figure 2 demonstrates the architecture of the gateway service including the various parts as described before. It includes the producers are the sensors, the gateway includes the front end services such as notification service, sensors data collector service, connectivity service, local storage and the application logic, the cloud (back end) includes the messaging service (Google cloud messaging). Also the cloud (back-end) includes the connectivity service, the identity management, the local storage (divided over two database systems such as the data historic values from sensors and JSON storage service), the identity management service and the application logic, and the consumers are the third party applications and services accessed in the form of web applications (e.g. from medical personnel, doctors or technicians).In next section we detail the roles and the operations for each of the modules of Figure 2.

3.2.1 Producers

Producers are the users with sensors that produce information including user (content) and sensor data (context data). Users (that in case of Figure 2 could be patients) use sensors and a mobile device to collect data that the sensors produce. Furthermore, users can utilize the front-end interface to choose optionally and on demand to send data to the back-end wherein third party users (e.g. doctors and physicians) can access it. Finally, they are able to receive updates or notifications when required (e.g. a situation is critical). To develop our solution, we utilized two types of sensors; the Polar H7[6] and the Onyx II[7] that include the following features. The polar H7 is a BLE transmitter recording in real time accurate the heart rate levels. The device is attached around the chest with an elastic belt and it detects heartbeat, and then it gives a timing reference for a specific heart rate measurement and transmits the information to back-end. The Onyx II (we used the model 3230) supports Bluetooth Smart wireless technology and provides an oximetry monitoring solution. It is a health monitoring device that allows patients, together with clinicians to easily monitor vital data such as the oxygen saturation range and heartbeats per minute. As a result, patients can perform daily activities and send vital data via wireless communication devices such as mobile phones and computers to Internet.

3.2.2 IoT Gateway

The gateway part consists of services that collect data produced by sensors over time, process and store data locally and send it to the cloud-back end. Current IoT gateway version is implemented for Android. The service allows registration of new users and configuration of sensors with rules per user. Also, it enables user notifications in the form of messages. The architecture of the gateway module consists of the following. The sensor data collector collects data and send it to the other sub-sections of the service. The local storage service supports specific data types by including an XML schema for all supported sensors, as well as identification of the users and rules. Finally, it includes the averaging measurements that the sensors produce at certain intervals. The notification service monitors the service conditions for possible changes in order to immediately notify the user. It also allows the user to transfer the data to the cloud on demand (e.g. when an emergency situation is sensed). The front-end connectivity service is the channel of communication between the gateway and cloud back end system. The data properly encoded to a JSON string form and is sent to the cloud-based on the HTTP protocol. The front-end application logic is the mediate system that combines aforementioned processes. It defines how services communicate and how the data is flown from front-end to back-end system and to the third party users. Data acquired from sensors are stored temporarily in local storage and are forwarded to the cloud at predefined time intervals (eg every 5 minutes). However, the user can activate an emergency operation which overtakes this operation by starting forwarding of all data to the cloud in real time

3.2.3 Cloud Back-End

The back-end system by special purpose services used to process and store the data of all sensors that interact with the system. In particular, these are (a) the Google cloud messaging[8], (b) the JSON storage GE, (c) the KeyRock identity management GE and (d) the back end application logic as the mediation among all processes. The back-end architecture consists of the following.

[6] http://www.polar.com

[7] http://www.nonin.com/Onyx9560

[8] https://developers.google.com/cloud-messaging/

a) The back-end connectivity service is the channel of communication. Here data is encoded in the JSON string form so to be forwarded to other parts.

(a) The storage service for storing information about user devices, sensors, rules related with the sensors, and historical data that is send from a sensor in conjunction with a date stamp. The storage service includes a JSON storage module as in [5] that supports data storage in JSON format based on a Mongo DB database[9]. Its main features are to create, update and delete users as well as the creation of databases, collections and records. The JSON storage service is a ideal for real time data storage. It also provides security features following the Oauth2 protocol[10].

b) The Google Cloud Messaging (CGM) service for sending messages asynchronously using as a an identifier of the device id for each user. We use this service since there is not a simliar FIWARE implementation. The service includes a Google server connection, a user server and a client application. First the service includes an application that sends an HTTP request to the GCM server, then the Google connection server responds to the message device and gives a unique registration identifier that characterizes it. Then the gateway service sends the registration identifier to the server for later use. Then the service uses the registration identifier along with the message text and makes a request to the GCM server so to send the message to the Client App. GCM server securely sends a message to the Client Up.

c) The identity management for user authentication (users register here to get access to the service).

d) The back-end application logic includes the controls, regulations and all the necessary API calls.

3.2.4 Consumers

Consumers are the third party users who access information produced by the systems and the service interface. The "Consumer" could be technical specialists that (a) define schemas for new sensors, (b) create rules for users and (c) send messages at regular intervals, or specialized users that can have access to historical measurements and monitor the real time data. Clinicians or other health careers are consumers too. So we can identify at least two types of consumers a) technicians and system administrators and b) health careers. The consumers part includes the following. The "Web application" is the user interface to interact effectively with the operations offered by the service.

4. DESCRIPTION OF THE SERVICE

This section presents the description of the XML schema for adding a new sensor, the sensor data collector, the local storage, the connectivity service, the notification service and the application logic as demonstrated in Figure 1. It should be mentioned that the current version supports Android[11] devices.

Firstly the XML schema to add new sensor is a generalized information model that accepts the recognition of attribute values of sensors based on the general characteristics profile (GAAT). In particular, the attribute value is a specific feature type address (attribute type) and characterized by a particular and unique universal unique identifier (UUID). Using such an XML schema with the appropriate tags we are able to dynamically insert new sensors and to identify values of their characteristics such as measurement of the heartbeat. The description of the XML schema includes the following information (a) <sensor>: This tag includes the BLE devices that are registered into the system, (b) <device>: A specific sensor which is characterized by its category (e.g. Polar H7), (c) <values>: Indicate the measurement type and the measurement units. For example, the H7 Polar device generates pulse in bpm unit, (d) <format>: It is the format of produced values (e.g. FORMAT_UINT8 that is an 8-bit integer) and (e) <position> & <multi>: In some cases, BLE sensor manufacturers define complicated function for decoding their features. The specific tags help us to manage the combination of the dynamic values by determining the location of the value within the attribute (<position>) and multiplying with a suitable integer value (<multi>).

The sensor data collector, collects data from the device based on the XML schema. It uses the "Device Scan Activity" that receives and stores an appropriate data structure of the XML schema for BLE sensors. Then, it activates the mechanism for detecting peripheral devices to determine which peripheral devices advertise data to the service. The sensors generate data per second, using a broadcast receiver, and information is forwarded to the application logic. The local storage allows insertion, deletion and update of data and it includes the database manager, the data encryption and decryption and the SLQLite database modules. The Connectivity Service is the communication channel between the gateway service and the cloud. Data is transformed to JSON string through the JSON parser and then with the suitable asynchronous calls are send securely to the cloud. The notification service informs the users with appropriate prompts namely "toast messages" for Android devices and "alerts" for important changes occurring in the flow of operation of the service. Its various operations include BLE device activation update, changing conditions depending on the individual case of the service, and (d) the update of the user for third party user messages. The application logic is the centralized module that orchestrates all services. The service includes the management module for comparing the measurements produced by the sensors and the rules stored in the cloud.

5. SERVICE EVALUATION

The performance evaluation of the IoT gateway service includes the calculation of the data transmission time thought the REST API calls between the service (front end) and the cloud system (back end). We also measure the signal processing and storing times by estimating the average time of 100 method executions. Firstly, we use the method "user registration" to insert a new and his/her gateway device to the cloud. The device requests a new "registration id" from the GCM server, an action that requires an average of 228 milliseconds (ms). Then the request is forwarded to the JSON storage service through a HTTP post request that takes an average of 140 ms and data is stored (an action that requires 4 ms). Based on this measurement, the whole process for this method is 372 ms. Secondly we use the method "get rules" to retrieve rules from the cloud through an HTTP post call that it takes an average of 274 ms, then the method stores data to SQLite database (4ms) of the gateway device. The total time is averagely calculated to 278ms (for the 100 method executions).

[9] https://www.mongodb.org

[10] http://oauth.net/2/

[11] https://www.android.com

The method called “data transmission” is responsible for transferring sensors data to the cloud using the gateway service. For experimental purposes we used the “Polar H7” and the “Nonin Onix 3250" sensors that generate new measures every 1 second. The service converts data to JSON format (2ms) and with asynchronous HTTP calls (120 ms) forwards it to the local storage where storage process takes around 6 ms. So the whole time process for the “data transmission” method is calculated to 128 ms. The final method is the “message notification” where messages are send through GCM to the device. To send a message, the third party user types submits the main text and the device’s registration id that is send to GCM server (400 ms), which in turn discovers the device for which the message should be forwarded (a process that takes 128 ms). The service displays this message as a notification to the user. For this method the whole processing time is about (528 ms). We can observe from aforementioned results that the data transmission time of 128 ms is significantly low for real time data. Similarly, the collection of rules from the cloud, an action that happens rarely e.g. when new rules are set, its again low (278 ms). The message notification total time is 528 ms, however this does not affect the data collection process, only the communication between users.

6. CONCLUSIONS

Cloud empowers IoT by providing an infrastructure to monetize resources for elastic data storage and to ease development of innovative applications for complex sensing data fusion and analysis. This works proposed an automated sensor data collection service for BLE devices. We presented a flexible and dynamic architecture for registering new BLE sensors and to conceptualize the data capturing and forwarding to the cloud. We developed the system based on general purpose services and we support functionalities such as a generic and dynamic registration of BLE sensors via an XML schema (thus support of semi structure data). The experimental section demonstrates an average low latency for the various processes of the service, however in next studies we target to explore further experiments for a higher number of sensors.The contribution of this work is on the IoT data transmission rate that is averagely calculated to 128 milliseconds and in the experimental section we discuss that this is significantly low for real time data. Also, we aim to support more mobile operating systems as the current prototype service is implemented for Android devices. As future extension we plan to explore real time data storage to the JSON based storage in the cloud in order to evaluate its performance and define capacity characteristics (i.e. how many users and devices can be hosted at the same time).

7. REFERENCES

[1] Zahariadis, T., Papadakis, A., Alvarez, F., Gonzalez, J., Lopez, F., Facca, F., and Al-Hazmi, Y. 2014. FIWARE Lab: Managing Resources and Services in a Cloud Federation Supporting Future Internet Applications. In Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC '14). IEEE Computer Society, Washington, DC, USA, 792-799. DOI=http://dx.doi.org/10.1109/UCC.2014.129

[2] Koreshoff, T. L., Robertson, T. and Leon, T. W. 2013. Internet of things: a review of literature and products. In Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration (OzCHI '13), Haifeng Shen, Ross Smith, Jeni Paay, Paul Calder, and Theodor Wyeld (Eds.). ACM, New York, NY, USA, 335-344.

[3] Sotiriadis, S., Petrakis, E.G.M., Covaci, S., Zampognaro, P., Georga, E., Thuemmler, C. 2013. "An architecture for designing Future Internet (FI) applications in sensitive domains: Expressing the software to data paradigm by utilizing hybrid cloud technology," in Bioinformatics and Bioengineering (BIBE), 2013 IEEE 13th International Conference on , vol., no., pp.1-6, 10-13 Nov. 2013

[4] Sotiriadis, S. and Bessis, N. 2016. An inter-cloud bridge system for heterogeneous cloud platforms. Future Gener. Comput. Syst. 54, C (January 2016), 180-194. DOI=http://dx.doi.org/10.1016/j.future.2015.02.005

[5] Preventis, A., Stravoskoufos, K., Sotiriadis, S. and Petrakis Petrakis, E.G.M. 2014. Interact: Gesture Recognition in the Cloud. In Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC '14). IEEE Computer Society, Washington, DC, USA, 501-502.

[6] Christopher J. Hansen. 2015. Internetworking with Bluetooth Low Energy. GetMobile: Mobile Comp. and Comm. 19, 2 (August 2015), 34-38.

[7] Want, R. 2015. The Physical Web. In Proceedings of the 2015 Workshop on IoT challenges in Mobile and Industrial Systems (IoT-Sys '15). ACM, New York, NY, USA, 1-1.

[8] Ikram, A., Anjum, A., Hill, R., Antonopoulos, N., Liu, L. and Sotiriadis, S. 2015. Approaching the Internet of things IoT: a modelling, analysis and abstraction framework. Concurr. Comput. : Pract. Exper. 27, 8 (June 2015), 1966-1984.

[9] Al-Salih, W. 2009. Mobile Data Collectors in Wireless Sensor Networks. Ph.D. Dissertation. Queen's University, Kingston, Ont., Canada, Canada. AAINR65438.

[10] Collotta, M. and Pau. G. 2015. Bluetooth for Internet of Things. Comput. Electr. Eng. 44, C (May 2015), 137-152. DOI=10.1016/j.compeleceng.2015.01.005 http://dx.doi.org/10.1016/j.compeleceng.2015.01.005

[11] Kravets, R., Tuncay, G. S., and Sundaram. H. 2015. For Your Eyes Only. In Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services (MCS '15). ACM, New York, NY, USA, 28-35. DOI=http://dx.doi.org/10.1145/2802130.2802137

[12] Miller, F.P., Vandome, A.F. and McBrewster, J. 2009. Advanced Encryption Standard. Alpha Press.

[13] European Commission. Definition of a research and innovation policy leveraging Cloud Computing and IoT combination. Tender specifications, SMART 2013/0037, 2013.

[14] Botta, A., de Donato, W., Persico, V., and Pescape, A. 2014. "On the Integration of Cloud Computing and Internet of Things," in Future Internet of Things and Cloud (FiCloud), 2014 International Conference on , vol., no., pp.23-30, 27-29 Aug. 2014