An IoT Gateway Centric Architecture to Provide Novel M2M Services

Soumya Kanti Datta, Christian Bonnet, Navid Nikaein
Mobile Communication Department
EURECOM
Biot, France
{dattas, bonnet, nikaeinn}@eurecom.fr

Abstract— **This paper proposes an innovative Internet of Things (IoT) architecture that allows real time interaction between mobile clients and smart/legacy things (sensors and actuators) via a wireless gateway. The novel services provided are: (i) dynamic discovery of M2M device and endpoints by the clients, (ii) managing connection with non-smart things connected over modbus, (iii) associate metadata to sensor and actuator measurements using Sensor Markup Language (SenML) representation and (iv) extending the current capabilities of SenML to support actuator control from mobile clients. These clients are equipped with an end-user application that initiates the discovery phase to learn about the devices and endpoints (sensors and actuators) connected to the wireless gateway. Then the user can select desired sensors to receive and display sensor metadata and control actuators from the mobile device. Prototypes of the mobile application and the wireless gateway have been implemented to validate the entire architecture. The gateway is implemented using RESTful web services and currently runs in a Google App Engine. Two real life scenarios are discussed that can be implemented using the architecture. Finally overall contributions and future research scopes are summarized.**

Keywords- M2M device & endpoint; Internet of Things; Wireless gateway; SenML; Mobile application.

I. INTRODUCTION

Recently there has been an increasing interest in smart devices containing endpoints like sensors and actuators also known as the things. They can connect to the internet to cooperate and create new services to citizens and different industries. The typical application scenarios include smart home, e-health, smart grid and more. The resulting eco system is referred to Internet of Things (IoT) [10], [11], [13]. The interconnection of the smart things is called machine to machine communication (M2M). It enables machines to communicate with each other without human interaction. An M2M device is a device that runs applications using M2M capabilities and network domain facilities. These devices may contain one or more endpoints. To achieve the mentioned eco system, the endpoints need to be connected to software and made available to be used together as a system. M2M architecture and protocols address the first item but the second one remains a challenge. This is due to the fact that these M2M devices are not capable to connect directly to the internet as there is lack of uniform standardization in communication protocols and sensing technologies. Also several other factors like limited computational power, memory and power resources of these devices prevent them from direct connection to internet. Therefore an intermediate device, called “gateway”, is typically employed between the network of the M2M devices and the remote peers (e.g. clients) over the internet. The goal of the gateway is (i) to settle with the heterogeneity between different endpoint networks and internet, (ii) strengthen the management of the endpoint networks, (iii) bridge traditional internet with endpoint networks.

Current literatures present several architectures on IoT. The authors of [15] have put forward an architecture comprising of sensing layer, IoT network layer and IoT application layer. They have also identified the key problems: formalization, standardization and data problem. Solutions for information security and data processing are presented. The paper [14] proposes a solution that allows service portability across systems. It is focused on the naming and addressing issues of connected objects, network elements supporting the APIs for application portability and related gateway architecture. The architecture is capable of supporting ubiquitous services on an end-to-end basis. The paper [9] deals with networks in sensing domain and describes the features of IoT gateway. Also a model gateway for IoT applications is proposed. Similar researches are explained in [17], [18]. Web IoT is also presented in [16] which outlines the use of web services in realizing IoT architecture. But these researches have not addressed some real life problems like device and service discovery, controlling endpoints from mobile clients and interaction with endpoints through standard protocol. Our work aims to provide solutions for them using a realistic architecture.

This paper introduces a wireless gateway (WG) that acts as the backbone of the proposed IoT architecture shown in Figure 1. The architecture is composed of three layers: (i) sensing layers containing the M2M devices and endpoints, (ii) gateway API layer and (iii) application layer. The M2M devices and endpoints must register themselves in the gateway to be made available to the mobile clients. They have no prior knowledge of the things. Thus they initiate the discovery phase by establishing a connection to WG. The discovery phase is also necessary due to the fact that M2M devices and endpoints can be added or removed at real time which the clients must learn in order to operate. The necessary API for the discovery phase is implemented in the gateway. This is an important and novel solution provided by the architecture. After the discovery phase, the user can choose desired sensors to receive the measurements which are represented using SenML [2]. It is lightweight and is designed keeping in mind the limited capabilities of M2M smart devices so that they can easily encode sensor measurements into different media types. Parsing SenML encoded data is very easy as it is implemented using JSON. This enables a server to efficiently parse huge numbers of sensor metadata at a very short span of time.

Another important problem is to manage the connection with the device containing legacy thing. It is incapable of generating the SenML metadata. Such device is connected using a serial communication protocol Modbus [3] to WG which converts the raw measurements into SenML metadata. The architecture also addresses actuator control from mobile clients. But the current implementation of SenML only supports sensor measurements. Thus we have extended the capabilities of SenML to address actuator control problem.

A prototype of the architecture has been implemented in which the APIs for WG run in a Google App Engine (GAE). The APIs are composed of RESTful web services. The mobile clients are equipped with an application named "Connect and Control Things" (CCT) that receives the measurements from sensors and can control actuators via the GAE. The SenML metadata for endpoints are implemented using JSON. Each component of the architecture is described in details along with the novel services in the following sections.

The rest of the paper starts with section II which focuses on the different components of the IoT architecture. The novel services offered by these components are explained. Section III highlights the prototypes. Section IV outlines two real life scenarios which can benefit from the architecture, particularly from the wireless gateway. Section V outlines our future research direction and concludes the paper respectively.

II. PROPOSED IOT ARCHITECTURE

A high level overview of the proposed architecture is depicted in Figure 1. The wireless gateway comprises of two interfaces: north and south. The north interface communicates with the mobiles clients and assists in discovery phase. The south interface manages and interacts with M2M devices (smart and non-smart) and stores their configuration in a local database.

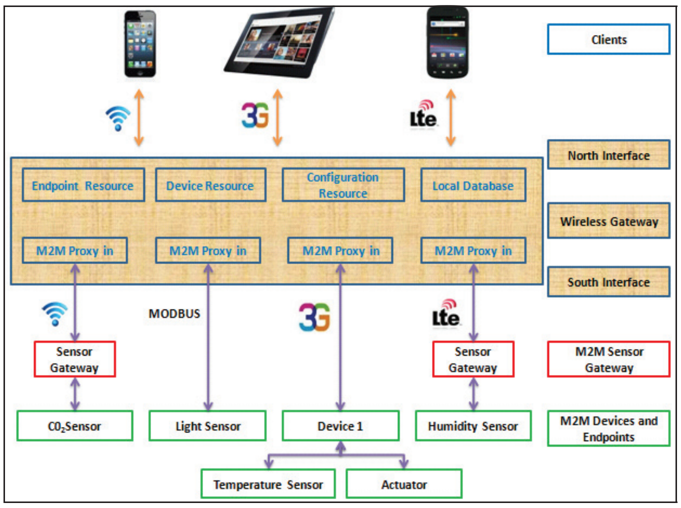


Figure 1. The proposed Internet of Things architecture.

A. M2M Devices and Endpoints

The actual sensing of physical entities (ambient light intensity, temperature) is done by different sensor endpoints which provide a raw measurement. In case of a smart M2M device, it locally associates a name, model number, hardware type, unit, version, type and timestamp to the sensor values. This creates a measurement metadata in SenML. Below is an example of a measurement from a light sensor encoded in JSON. It is seen that the array in the object has single measurement from the light sensor named "LS-1" with a value 200 lux. The time stamp is calculated with respect to 16/01/1970 AD at 23:34:57 UTC. The model and hardware type are optional.

```
{"e":[{"n":"LS-1", "v": 200, "u": "lx", "t": "1380897199", "ver": "2.0", "type": "Illuminance"}]]
```

The interface of the smart M2M device allows the sensor metadata to be read by a GET request or it can be pushed to the south interface of WG. For example, "Device 1" in Figure 1 is smart. For a non-smart M2M device (containing legacy endpoints), the measurements remain in the raw format and cannot be read by GET request. They are sent to the wireless gateway over Modbus. The M2M proxy-in invokes an API which creates the desired SenML compliant metadata. An intermediate sensor gateway (ISG) may also be present to assist the legacy device by creating the metadata and forwarding it to WG. In these cases, ISG and proxy-in are configured with the name, model number, hardware type, unit, version, type of the legacy device and timestamp to generate the metadata.

It is necessary to keep a record of the devices and endpoints for the discovery phase. Thus addition or deletion of devices and endpoints notifies the wireless gateway which updates its internal database and configuration accordingly. When a M2M device is added for the first time, it pushes its configuration list to WG where it is stored in the local database.

The main contributions here are twofold. Firstly, instead of reporting only the sensor value, a metadata is generated. The gateway assigns unique ID to each M2M device. The metadata expands the capabilities of the device due to the additional information. Secondly, non-smart devices can be managed using the architecture with the help of ISG.

B. Web Services of Wireless Gateway

The M2M services provided by the architecture are structured around wireless gateway. These novel services are designed and implemented as a web application based on the Representational State Transfer (REST) paradigm. Several reasons have motivated the choice: (i) ease of development, maintenance and huge popularity, (ii) proven interaction model for wide range of applications, (iii) SenML metadata can be carried as payload of HTTP and (iv) having direct access to M2M devices through web interactions. The gateway contains a central database to store the resource configurations of devices, endpoints and sensor metadata. The functionalities and APIs are broadly associated with the two interfaces, north and south.

The north interface directly interacts with the mobile clients and its main responsibility is to assist clients in device and endpoint discovery. It also communicates capabilities of the actuators to the clients for actuator control. The functionalities of the interface are explained below.

- **Device and endpoint discovery**: In real time scenarios, the clients do not have any prior knowledge of the M2M devices and endpoints attached to the wireless gateway. Initially the client establishes a connection to WG and initiates the discovery phase. The north interface then queries the local database containing the list of M2M devices, their endpoints and corresponding configurations. The entire list is communicated back to the client. It also learns about the configurations of actuators during the phase. Thus the north interface provides a novel service in terms of dynamic discovery. After the discovery phase is over, if a device is added to or removed, the database of the gateway is updated accordingly and the clients receive a push notification of the same.

- **Reporting sensor metadata**: After the discovery phase, the user selects desired endpoints (sensors) to receive the metadata. The client sends a GET request to the north interface which interacts with the web services at south interface to collect the metadata and sends it back to the clients.

The south interface is a collection of REST web services divided into two categories, namely proxy-out and proxy-in.

- **Proxy-in**: An M2M device is primarily connected to a proxy-in which registers the former with its endpoints and collects sensor metadata. When the device is added to a zone for the first time, it sends the configuration to the proxy-in which adds an entry to the configuration database in the wireless gateway. Proxy-in can send a GET request to the connected M2M device to receive the metadata and send it to the clients. The device can also push the metadata to proxy-in which in turn pushes the same to the clients. Thus proxy-in facilitates real time interaction between clients and endpoints. In case of non-smart device, there is a protocol to translate the raw data into SenML format. The south interface can also store the metadata in local database and the north interface can retrieve the metadata from the database and send to the clients.

- **Proxy-out**: A proxy-out links clients at the north interface with the M2M devices. The requests from the clients are carried over HTTP. The proxy-out receives the location of the M2M device as the path of the HTTP request. There is a protocol that translates the HTTP payload into M2M device specific command which the endpoints will understand. The proxy-out may need to interact with a non-smart device. In that case the instruction received from the client is converted into machine executable format by suitable mapping to a local database entry. Following example illustrates the functionality of the proxy-out.

  ```
  Request from client
  PUT /proxyout1.mydomain.com/dev1.mynetwork.com
   <senml bn=urn:dev:mac:6188322>
  <e n=”Temp-1” t=”0” v=”15” u=”Cel” />
   </senml>
  ```

  Response

  204 no content

  In the above example, the client requests the proxy-out located at “proxyout1.mydomain.com” to connect to the device at “dev1.mynetwork.com” having MAC address “6188322” to set the temperature to 15 degree Celsius. The proxy-out replies with HTTP status code 204 with no content.

An important innovation in the wireless gateway is to extend the current capabilities of SenML to support actuator control from clients. According to the work-in-progress draft of SenML [6], it only carries the sensor metadata in HTTP SenML MIME type and does not allow pushing commands to the actuators. We propose add that support which enables the clients to manage the actuators along with sensors with minimum efforts. We have introduced several features to manage actuators using the same SenML implementation. The extensions are implemented in the wireless gateway web services and are explained below.

- **Provision for actuator**: In order to be able to control an actuator from the client, the provision for actuator must be added. This is done by adding another semantic for actuator to the unit resource type description in SenML. A unit resource is a semantic field linked to a SenML unit. Currently it is set to sensor but with the proposed addition actuators can be addressed using same SenML implementation. This allows the client to differentiate between a sensor and an actuator since both endpoints could be attached to the same M2M device.

- **Configurations of actuator**: To interact with an actuator, its configurations must be known to the client. The list includes the following: (i) type of actuator, (ii) supported functionalities (e.g. for controlling a light – switch it on/off and dim it), (iii) allowed range of values – may be continuous or discreet (e.g. for controlling an air conditioner, the range of temperature), (iv) unit and (v) location. The SenML draft is enriched with these additional capabilities of actuator. These configurations are reported to WG when the actuator connects to WG for the first time.

- **User interface at the client**: It is necessary to associate different measurements with different representations at the client. We propose to drive the user interface representation from the wireless gateway using different images for different measurements.

These extensions are supported by the APIs of wireless gateway and provide a solution of real life problem of controlling actuators from clients.

C. *Mobile Clients*

The mobile clients included smartphones and tablets running Android and iOS. These clients are running CCT that performs the following: (i) dynamic discovery of M2M devices and endpoints, (ii) receive and display sensor measurements, (iii) receive push notifications from the gateway and (iv) controlling actuators. Each representation is associated with a pre-defined range of values of sensors. In order to cater maximum number of clients with different mobile platform, the application has been developed using cross platform mobile application development tools.

III. PROTOTYPES IMPLEMENTATION

Prototypes of CCT and web services of WG are implemented to validate the proposed architecture. The prototype has been tested using real time sensor metadata coming from an M2M device remotely deployed. At the same time, there is an option to simulate the M2M device and endpoints at the wireless gateway. Following subsections describe the prototypes.

A. Mobile Application: Connecting and Control Things

CCT is developed using cross platform tools PhoneGap 2.9.0 and JQuery Mobile 1.3.1. The reasons behind choosing cross platform mobile application development approach over native approach are: (i) cross platform allows developing application for multiple mobile platforms and (ii) application development time and cost minimize to a great extent [6]. We have developed CCT for Android and iOS devices.

Initially CCT connects to the GAE server having address [1] where the web services of the wireless gateway are running. Once the connection is established, the application initiates the discovery phase and learns about the M2M devices and endpoints. Then a device can be selected to view the endpoints. Following is a screenshot of CCT depicting an M2M device with a temperature sensor and an actuator for temperature control as endpoints.

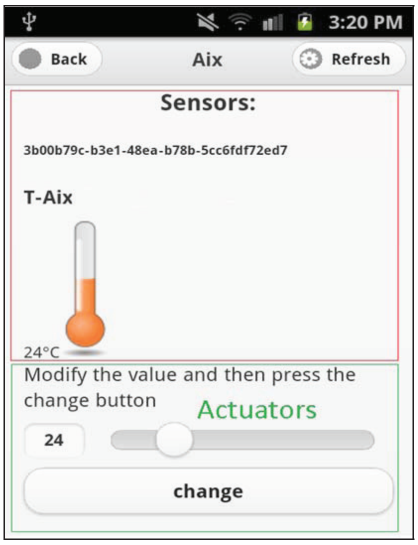

3:20 PM
Back
Aix
Refresh
Sensors:
3b00b79c-b3e1-48ea-b78b-5cc6fdf72ed7
T-Aix
24°C
Modify the value and then press the change button
Actuators
24
change

Figure 2. An M2M device with two endpoints.

The actuator can be used to control the temperature. When the temperature value changes, the application updates the UI dynamically as seen from Figure 3.

There are following two ways to obtain the sensor metadata.

- **Polling**: - In this case, CCT sends GET request to GAE and the reply contains the sensor measurements and metadata.

- **Pushing**: - This is implemented to notify the clients about sensor reading update when CCT is in background. To receive such notification, the applications from Android devices must register itself in Google Cloud Messaging (GCM) service which gives each client a unique ID (UID). The UID is then communicated to the GAE. When there is a sensor update received at the south interface, the GAE sends the notification and the UID to GCM. It pushes the notification to the client corresponding to the UID. For the iOS clients, Apple Push Notification Service has to be used instead of GCM and rest of the mechanism remains the same.

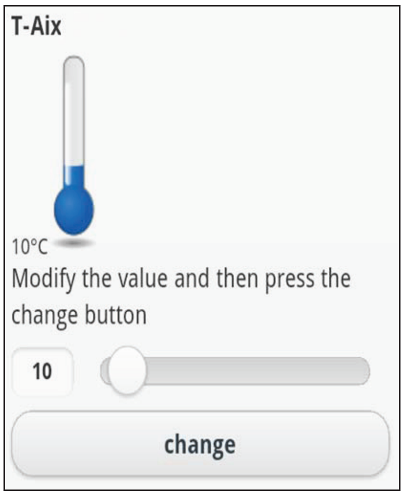


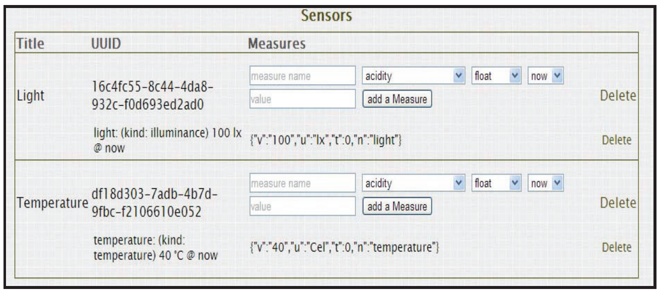
Figure 3. Setting a new temperature using the actuator.

B. Google App Engine for the Wireless Gateway

The web services of WG are implemented using Java and are running in a Google App Engine (GAE). GAE server provides the necessary storage, services and other infrastructure to host and run the APIs.

There is an option to simulate M2M devices with endpoints in the GAE server. It allows simulating several devices in different zones and subzones. The values of the endpoint, unit, name, type, timestamp can be configured from the zone admin page. For actuator, the range of permitted values is to be given. A screenshot of two simulated sensors is given in Figure 4. It is seen that the server assigns UID to the sensors. The back end processing is capable of storing the sensor configuration in local database, generating the SenML metadata and sending it to CCT. If the client has registered for push notification, the new metadata can be pushed to the client.

The south interface is also implemented and is capable of receiving data from real M2M devices. Such a device with a sensor is deployed remotely and the sensor metadata is sent from the device to the south interface of the wireless gateway running in the GAE. The metadata is then sent to the clients.



Sensors

Title	UUID	Measures	
Light	16c4fc55-8c44-4da8-932c-f0d693ed2ad0	measure name / acidity / float / now / value / add a Measure	Delete
	light: (kind: illuminance) 100 lx @ now	{"v":"100","u":"lx","t":0,"n":"light"}	Delete
Temperature	df18d303-7adb-4b7d-9fbc-f2106610e052	measure name / acidity / float / now / value / add a Measure	Delete
	temperature: (kind: temperature) 40 °C @ now	{"v":"40","u":"Cel","t":0,"n":"temperature"}	Delete

Figure 4. Simulating new sensors at the Google App Engine.

C. *Use case with a sensor as M2M endpoint*

A use case is implemented putting together all the components of the architecture. A Tablet running Android 4.0.4 is used as client that runs CCT. An M2M device containing a temperature sensor is used. The steps are sequentially presented in Figure 5.

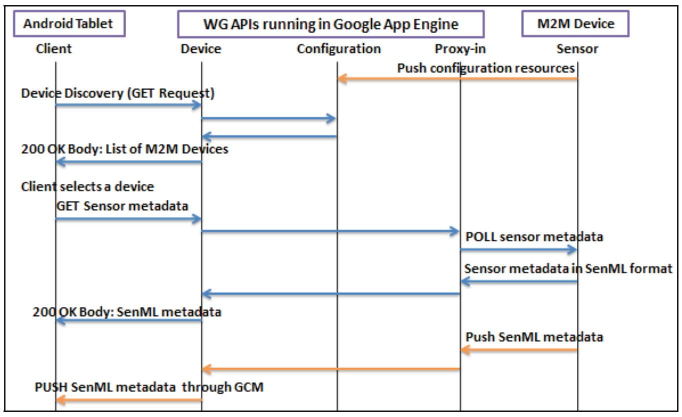

Android Tablet
WG APIs running in Google App Engine
M2M Device
Client
Device
Configuration
Proxy-in
Sensor
Push configuration resources
Device Discovery (GET Request)
200 OK Body: List of M2M Devices
Client selects a device
GET Sensor metadata
POLL sensor metadata
Sensor metadata in SenML format
200 OK Body: SenML metadata
Push SenML metadata
PUSH SenML metadata through GCM

Figure 5. Use case with a sensor.

The APIs for device, configuration and proxy-in are running in GAE. Initially the M2M device pushes its configuration in the gateway database. Then the client connects to the north interface (Device API) for discovery. Once the desired sensor is chosen, a GET request is sent from CCT to gateway. The proxy-in at the south interface polls the sensor metadata and using the north interface the metadata is sent to CCT. The client is also subscribed to the push notification. Thus when there is a new sensor reading, it is pushed to the south interface and WG pushes a notification with the update metadata through GCM service. Since there is no actuator involved, proxy-out is not used at the south interface.

D. *Use case with an actuator as M2M endpoint*

Another use case is implemented using an actuator as M2M endpoint. The scenario is portrayed in Figure 6.

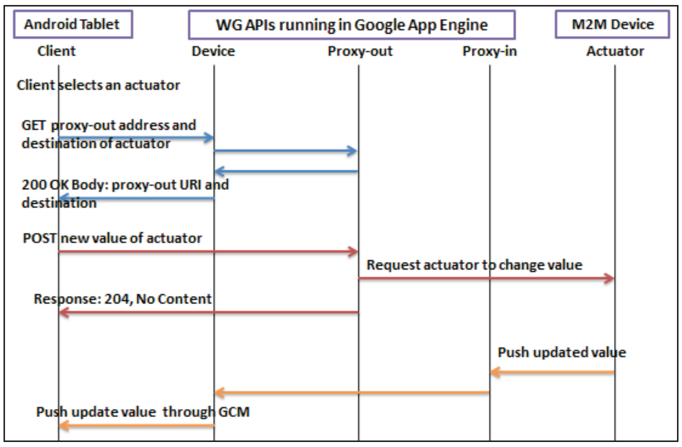

Android Tablet
WG APIs running in Google App Engine
M2M Device
Client
Device
Proxy-out
Proxy-in
Actuator
Client selects an actuator
GET proxy-out address and destination of actuator
200 OK Body: proxy-out URI and destination
POST new value of actuator
Request actuator to change value
Response: 204, No Content
Push updated value
Push update value through GCM

Figure 6. Use case with an actuator.

Figure 6 skips initial steps like discovery. From CCT, the user directly selects an actuator and it sends a query to the north interface to learn the proxy-out address (which interacts with the actuator) and destination (URI of M2M device containing the actuator). Then the user can select a value within the permitted range and send a command to update the value using POST. The proxy-out receives the command, translates it into a suitable format for the actuator and requests it to change the value. Also the former sends back HTTP status 204 to CCT to indicate that the request is forwarded to the actuator. After the value is updated, the new value is sent to the client using the push mechanism.

IV. USAGE OF THE ARCHITECTURE

Several novel services can be implemented based on the proposed IoT architecture. The paper discusses following two types of application scenarios.

A. *Personal health monitoring*

Such implementation of the mobile client and wireless gateway is highly applicable in personal health monitoring. Consider a scenario in which the M2M devices are composed of pacemaker, thermometer, and blood pressure devices. These devices periodically monitor a person's health, convert the sensor values to SenML format and push to the WL-Box. At the client side, a doctor can connect to the WL-Box from a smartphone or tablet. The application can use the discovery phase to learn how many patients are being monitored and select a patient to read the vitals. Since the application represents the sensor data graphically, a doctor can better understand the state of the vitals from the display. If further check-up is necessary, the doctor can contact the patient with proper advice. Thus health problems can be avoided. Similar solution has been proposed in [7].

B. *Adding semantics to sensor measurements*

The prototypes can be used to realize the research on semantics presented in [4], [5]. It uses a similar architecture to merge heterogeneous sensor networks and used ontology to automatically convert sensor data into semantic data. Then a genetic algorithm is used to reason on the semantic data. The only difference in the designed architecture is the usage of

M2M aggregation gateways. They actually add semantics to the sensor measurements. This research paves way for an innovative application which can suggest a recipe based on the food available in the kitchen, adapted to the surrounding weather and the recipe is tailored to personal diet.

V. CONCLUSION AND FUTURE DIRECTION

The internet of things is being termed as the future of internet communication [12]. This work is an attempt to integrate sensors and actuators with web services and wireless gateway that could fit in most common small devices today. We have described a general IoT architecture to provide novel M2M services. The mobile clients interact with the M2M devices and endpoints through the gateway. The M2M services including device discovery, local storage of resource configurations are structured around the gateway. The architecture is capable of interacting with a non-smart device over Modbus. Prototypes of the mobile application and WG are discussed. Such realistic architecture can be used in several real life scenarios including personal health monitoring, environmental monitoring and semantics in IoT. The usefulness of the system lies in the fact that it provides an incremental methodology to add new services and is generic enough to be compliant with future standards in M2M communications and IoT. The system is scalable to handle huge amount of traffic as it is based on RESTful paradigm and SenML is very lightweight which can be processed very fast. As future direction, the gateway interfaces and their functionalities are being implemented using IPSO Alliance Framework [8], access rights for multiple users and security & privacy of IoT architecture are being investigated.

ACKNOWLEDGMENT

The authors would like to thank Ms. Isabelle Dalmasso and Mr. Lorenzo Sfarra for their contributions to the research. The work is sponsored by French research project “WL-Box 4G” Pole SCS.

REFERENCES

[1] http://emulator-box-services.appspot.com/senmladmin/zones

[2] http://tools.ietf.org/html/draft-jennings-senml-10

[3] http://en.wikipedia.org/wiki/Modbus

[4] A. Gyrard, C. Bonnet and K. Boudaoud, “An architecture to aggregate heterogeneous and semantic sensed data, ” in 10th Extended Semantic Web Conference, PhD Symposium, (ESWC 2013), May 26-30, 2013, Montpellier, France.

[5] A. Gyrard, C. Bonnet and K. Boudaoud, “A machine-to-machine architecture to merge semantic sensor measurements, ” in 22nd International World Wide Web Conference (WWW 2013), May 2013, Rio de Janeiro, Brazil.

[6] I. Dalmasso, S. K. Datta, C. Bonnet and N. Nikaein, “Survey, Comparison and Evaluation of Cross Platform Mobile Application Development Tools”, 9th International Wireless Communication and Mobile Computing Conference (IWCMC 2013), July 2013, Sardinia, Italy, in press.

[7] Yu-Hsien Chu; Yen-Chou Hsieh; Chia-Hui Wang; Yu-Chun Pan; Ray-I Chang, "UPHSM: Ubiquitous personal health surveillance and management system via WSN agent on open source smartphone," e-Health Networking Applications and Services (Healthcom), 2011 13th IEEE International Conference on , vol., no., pp.60,63, 13-15 June 2011

[8] http://www.ipso-alliance.org/wp-content/media/draft-ipso-app-framework-04.pdf

[9] Hao Chen; Xueqin Jia; Heng Li, "A brief introduction to IoT gateway," Communication Technology and Application (ICCTA 2011), IET International Conference on , vol., no., pp.610,613, 14-16 Oct. 2011

[10] Handong Zhang; Lin Zhu, "Internet of Things: Key technology, architecture and challenging problems," Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on , vol.4, pp.507,512, 10-12 June 2011

[11] Zhihong Yang; Yufeng Peng; Yingzhao Yue; Xiaobo Wang; Yu Yang; Wenji Liu, "Study and application on the architecture and key technologies for IOT," Multimedia Technology (ICMT), 2011 International Conference on , pp.747,751, 26-28 July 2011

[12] Khan, R.; Khan, S.U.; Zaheer, R.; Khan, S., "Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges," Frontiers of Information Technology (FIT), 2012 10th International Conference on , pp.257,260, 17-19 Dec. 2012

[13] Perera, C, Zaslavsky, A, Christen, P et al 2012, 'CA4IOT: Context Awareness for Internet of Things', Green Computing and Communications (GreenCom) 2012, IEEE and ACM (USA), Fance, pp. 775-782.

[14] Gronbaek, I., "Architecture for the Internet of Things (IoT): API and Interconnect," *Sensor Technologies and Applications, 2008. SENSORCOMM '08. Second International Conference on*, vol., no., pp.802,807, 25-31 Aug. 2008.

[15] Handong Zhang; Lin Zhu, "Internet of Things: Key technology, architecture and challenging problems," *Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on* , vol.4, no., pp.507,512, 10-12 June 2011.

[16] Castellani, A.P.; Dissegna, M.; Bui, N.; Zorzi, M., "WebIoT: A web application framework for the internet of things," Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE , vol., no., pp.202,207, 1-1 April 2012.

[17] Qian Zhu; Ruicong Wang; Qi Chen; Yan Liu; Weijun Qin, "IOT Gateway: BridgingWireless Sensor Networks into Internet of Things," *Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP 8th International Conference on* , vol., no., pp.347,352, 11-13 Dec. 2010.

[18] Costantino, L.; Buonaccorsi, N.; Cicconetti, C.; Mambrini, R., "Performance analysis of an LTE gateway for the IoT," *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a* , vol., no., pp.1,6, 25-28 June 2012.