Connect and Control Things: Integrating Lightweight IoT Framework into a Mobile Application

Soumya Kanti Datta and Christian Bonnet
Mobile Communications Department, Eurecom
Biot, France
Emails: {dattas, bonnet}@eurecom.fr

***Abstract*—Mobile applications running on smart devices are important enablers for Internet of Things (IoT). This paper presents a novel mobile application called "Connect and Control Things" (CCT) which integrates a lightweight IoT framework into smartphones and tablets. The framework is designed to discover & interact with things, reason with M2M data, provide configuration management facilities along with subscription and notification functionalities. The novel capabilities of CCT are: (i) lightweight search engine based dynamic discovery of physical things, (ii) interaction with both sensors and actuators using Sensor Markup Language (SenML) and its extensions, (iii) integrating a lightweight reasoning engine to infer high level abstraction, (iv) CoRE Link Format based configuration management and (v) support for both smart and legacy things regardless of communication technologies. An IoT architecture is presented along with a prototype implementation details for Android powered devices. Uses of the prototype mobile application in home automation scenarios are outlined. The performance of CCT is evaluated in terms of memory requirement, CPU usage and power consumption. Experimental results establish that the overall design and implementation of CCT are lightweight which encourages the adoption of CCT. It enables offering consumer centric services for home automation domain. Finally the paper concludes with future research directions.**

Keywords—Android; Configuration management; Discovery; Home automation; IoT framework; Mobile application.

I. INTRODUCTION

The Internet of Things (IoT) envisions a truly connected society by extending the Internet connection to physical things. It is estimated that by 2020, 50 Billion of things [13] will be connected to the Internet to create value added services for consumers. The development in the IoT ecosystem is further fuelling the growth of Machine-to-Machine (M2M) communications [1] [11]. Today the applications of M2M and IoT encompass a variety of domains including home automation, eHealth, smart tourism, smart agriculture and intelligent transportation system. A plethora of information and communication technologies (ICT) like cloud & edge computing, low power communication protocols, low cost sensing and actuating devices, smartphone & tablet applications are acting as enablers of the IoT ecosystem. The mobile applications are quite indispensable in terms of offering consumer centric services in IoT ecosystem. For example, in a simple home automation scenario, a user could create a custom access control policy for guest who will have limited access to the home automation devices. Also the smart devices now-a-days are equipped with high end hardware components, multiple sensors, multiple communication technologies, cutting edge operating systems and sophisticated software development kits (SDK). This creates a perfect environment for designing a lightweight IoT framework that can be integrated into smart devices as a mobile application. This would reduce the dependency on home gateways for protocol translation, IP communication, discovery and management of physical things.

Following the above motivations, this paper presents a lightweight IoT framework that is integrated into a mobile application called "Connect and Control Things" (CCT). The framework enables real time interaction with sensors and actuators allowing uniform metadata exchange using SenML [12] and its extensions [2] [3]. The framework can interact with things over multiple communication technologies supported by a smart device. Since the software drivers are already included in the operating system (OS), the driver installation and update are taken care by the OS itself. This further reduces the development time. The configuration management module keeps track of the thing configurations and allows the consumers to add, update and delete any configuration. The management module implements Open Mobile Alliance Lightweight M2M (OMA LwM2M) specifications [13]. To discover the things at real time, the framework employs a lightweight search engine. The discovery module is dependent on the management module to provide the "look up" facility and upon discovery, URI(s) corresponding to thing(s) are returned to the requestor. Finally, the framework also incorporates a lightweight reasoning engine for M2M data processing. The engine uses semantic web technologies to reason on raw sensor measurement and generates high level abstraction which on further processing turns into actionable intelligence. The derived intelligence allows the consumer or other M2M applications reacting to the environment using actuators. Along with that, the framework also provides subscription and notification services to the consumers. The main contribution of the paper is in integrating all these IoT functionalities into a mobile application.

The rest of the paper is organized as follows. Section II discusses the requirements of the IoT framework for developing CCT. Section III focused on the framework itself while Section IV describes prototype implementation details with experimental results. Section V highlights how CCT accomplishes home automation scenarios with Section VI concluding the paper.

II. REQUIREMENT ANALYSIS

The requirements for the IoT framework are analysed and summarized below.

- **Uniform data exchange with things**: The framework should be able to exchange data with heterogeneous things in a uniform and structured manner.
- **Real time interaction**: Consumer services must be offered in real time and should support multiple communication technologies.
- **Configuration management**: To keep track of the things, the framework must provide mechanisms for configuration management. The consumers should also be able to add, update and delete any configuration.
- **Dynamic discovery**: This is a fundamental requirement of any IoT framework. This module should enable discovery of things, their capabilities & properties and URIs to access them directly.
- **M2M data processing**: The framework must provide M2M data processing facility since raw sensor measurements does not convey any significant information.
- **M2M data storage and retrieval:** Appropriate data storage and retrieval mechanisms must be supported to store high level abstractions and other necessary events.
- Both smart and legacy devices must be supported. The legacy devices can be assisted using proxies [2].
- The system must allow session less interaction with consumer components.
- The functionalities of the IoT framework should be exposed to the consumer through RESTful web services.
- The framework must reuse the services (IP communication and location) offered by current network operators.
- **Subscription and notification:** The framework must allow the consumer to subscribe to certain events. Given that subscription, occurrence of the events should be sent as push notification to the consumers.
- **Access control:** Proper access control should be placed by the framework to limit the access to authorized services and things.
- **Interoperability:** The implementation of the modules should be interoperable among each other while the overall framework should be interoperable with similar IoT framework and platforms.
- **Security and privacy:** The framework must ensure end-to-end security to preserve privacy of consumers [8].
- **Consumer centric design:** To promote faster adoption among consumers, the framework must ensure low latency, high QoS, easy user interface and social network integration. Usability and recovery from are also highly necessary for such systems.

The requirement list is non-exhaustive but summarizes the main requirements for the framework. The framework is designed taking into consideration the above list.

III. Proposed Lightweight IoT Framework

The overall framework containing all the modules mentioned before is depicted in Fig. 1. The framework can be broadly classified into seven modules – (i) resource discovery, (ii) configuration management, (iii) metadata exchange, (iv) subscription and notification, (v) M2M data processing, (vi) data storage and retrieval and (vii) proxy manager. We assume proper access control policies are in place along with end-to-end security mechanisms. The framework interacts with physical things through a proxy layer at south interface while the functionalities for consumer smart devices are exposed through north interface. Both the interfaces are implemented using RESTful web services.

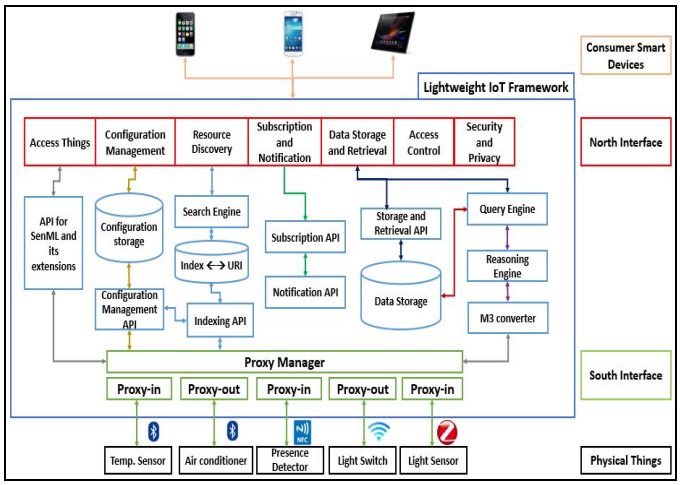

Fig. 1. Architecture of proposed IoT framework.

A. Proxy manager

The proxy manager is used to manage the communication with underlying things. The sensors and actuators are connected to proxy-in and proxy-out respectively [2] [4]. For each communication technology, there are dedicated proxy-in and proxy-out. The proxy manager contains the drivers necessary for technologies like Bluetooth Low Energy (BLE), NFC etc. Each proxy is implemented using RESTful web service. Therefore they can obtain sensor data using HTTP or CoAP [14] GET method or receive a push notification from things over HTTP or CoAP POST method. The manager provides protocol translation where necessary. Another novel feature of the proxy manager is that it can support legacy things. In this case, the manager could be configured with the properties of a legacy thing. For example, if a legacy temperature sensor is considered, then the corresponding proxy-in receives only the temperature measurement. But the proxy-in could be configured to add unit, timestamp, type of measurement, domain along with description of the legacy thing. This allows the framework to operate with both smart and legacy things.

B. Metadata exchange module

A fundamental aspect of M2M communication is the exchange of metadata generated by various things. SenML is used to represent the metadata which consists of sensor measurement along with additional information about unit, type of sensor, name, timestamp etc. The capabilities of SenML have been extended so that it can be used for metadata exchange with actuators also [3]. This is a novel feature of the framework where one generic API allows accessing things for metadata exchange. This functionality is exposed to the consumers through access things service at the north interface. In case of legacy sensors, SenML metadata is generated at proxy manager. For legacy actuators, the manager provides the protocol translation and converts the action into machine understandable format.

C. Configuration management

Another important building block of the framework is configuration management module. This supports registration and un-registration services for things. These services together with self-management service keeps track of the things connected to the framework. Also, the configurations of the connected things are managed by the module. When things connect to the south interface for the first time, they must upload their configuration. It is implemented using CoRE Link Format [4]. The configuration management API converts the received descriptions of things into appropriate format for configuration storage. The consumers can access the configuration of authorized things through configuration management service at the north interface. This allows the consumers adding, updating or removing thing description(s). For legacy devices, their descriptions are generated by the corresponding proxy.

D. Discovery module

IoT envisions interconnecting huge volume of heterogeneous things. This makes discovery a fundamental requirement for any IoT framework and platform. Discovery mechanisms [6] [7] are necessary to search for resources (things or services), their capabilities, properties and URIs to access them. The indexing API, index to URI databased and search engine are the components of the proposed discovery module. It is also dependent on the configuration management module. The configurations of the connected things are indexed using the indexing API to ease the discovery process. The index is unique to each physical thing. The index and an URI to access the thing are stored in a separate database. The discovery functionalities are exposed through resource discovery service at the north interface. When a consumer requests for discovery, the input for the request is analysed and keywords are extracted from that. Those keywords are used by the search engine to intelligently search across the stored configurations and indices. Upon a match being found, the URI along with the capabilities and properties of the thing (e.g. description of the thing) is returned to the requestor.

E. M2M data processing

The raw measurement coming from sensors does not convey much information. It could limit the scope of M2M applications if not processed. This paper proposes to utilize semantic web technologies to generate high level abstraction from raw sensor data. This is achieved in two steps. The first step is to add additional attributes to sensor measurement using SenML creating a metadata. The second step is to link the meaning of the data from the point of view of the domain in which the sensor is used. The generated high level abstraction of the sensor data is treated further to create actionable intelligence. This enables the consumers or other M2M applications react to the environment using actuators.

This module utilizes a lightweight version of Machine-to-Machine Measurement (M3) framework [5] [18]. The entire framework cannot be ported inside a mobile application because the former houses around 200 ontologies and related datasets, SPARQL queries. Instead, the lightweight version downloads the necessary ontologies, datasets based on the sensors attached to the south interface and the domains in which they are used. This is done during a provisioning phase. After that, the SenML sensor metadata is fed to the M3 converter which transforms the metadata into RDF format. That is then treated with a semantic reasoning engine and query engine to derive the actionable intelligence. The main novel aspect of this module is the integration of a lightweight semantic reasoning engine into a mobile application.

F. Subscription and notification

This module offers consumer centric services through north interface and allows the consumers to subscribe to certain events in the IoT framework. The subscription API keeps a unique ID for each consumer smart device. Upon occurrence of the corresponding event, a push notification is sent to the device with the unique ID using the notification API. For example, a consumer can subscribe to receive a notification when a new thing is added or an existing thing is detached from the south interface.

G. Data storage and retrieval

Finally the data storage and retrieval module is composed of an API and a database. The actionable intelligence derived from M2M data processing is stored at the data storage along with a timestamp. This could be later retrieved by the consumer by sending GET request to the corresponding service at the north interface. The storage is also useful to calculate statistical information of the M2M data. This module also provides data management functionalities to the consumers.

The above modules are designed to be interoperable among each other. This in turn makes the entire framework interoperable with similar IoT frameworks deployed in M2M gateways or cloud systems. The modules are also considered as common service functions (CSFs) which are a part of common service entity (CSE) in oneM2M architecture [15]. Therefore the design of the IoT framework and its integration in CCT are done following oneM2M specifications.

IV. Prototype Implementation

The above modules and the entire framework is successfully integrated into an Android application running on smartphones and tablets. This is a major contribution of the paper since CCT becomes an IoT hub that could be deployed in

any Android powered devices. The key implementation details for each module are provided below.

A. *Implementation details*

- **Proxy manager**: The Android operating system supports communications using BLE, NFC and Wi-Fi. The Linux kernel already provides the device drivers for these technologies. The CCT prototype can connect to things using these three technologies at the south interface. The proxy manager therefore creates and maintains an M2M local area network with the connected things.
- **Metadata exchange**: The API for metadata exchange is based on SenML and its extensions for actuators. In CCT the API is developed using JSON. Smart things can generated the metadata in JSON while the proxy manager can create the same for legacy things. JSON is transported as payload for HTTP based interactions with consumer devices.
- **Configuration management**: The things are described using CoRE Link Format which is implemented using JSON. This promotes interoperability among the IoT framework modules. It also reduces the coding complexity since JSON parser is readily available in Android SDK. The configuration storage API is basically converting the CoRE Link Format based description in SQLite storage.
- **Discovery module**: The configuration indexing is achieved using an indexing algorithm while SQLite database is used to store the index and URI. A lightweight library is written in Java to integrate the search engine functionalities. The reply of the discovery consists of thing description and an URI to access the thing. Therefore the reply is basically another JSON payload embedded into HTTP.
- **M2M data processing**: Originally, the M3 framework is developed using Apache Jena Framework. Since it cannot be ported to Android, AndroJena[1], a lightweight version of the Jena Framework is used. AndroJena is integrated as a library which allows implementing the reasoning engine in Java. Similarly the M3 converter and query engines are developed as separate APIs.
- **Subscription and notification:** This is done using Google Cloud Messaging (GCM) framework. The consumer can view a list of events for which subscription and notification services are offered and the steps are explained below using Fig. 2.

1. The Android powered client registers itself to the GCM framework to receive push notification.
2. The GCM assigns a unique ID to the client.
3. The client forwards the received unique ID to the subscription API of CCT along with the event for which the subscription is requested.
4. When the event occurs, the notification API retrieves the unique ID from subscription API. Then it sends an HTTP POST to GCM with notification and list unique ID.
5. Finally, GCM pushes the notification to the Android client corresponding to the unique ID.

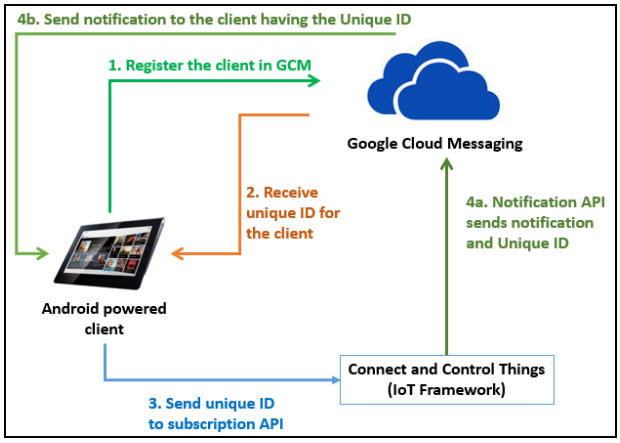

Fig. 2. Subscription and notification services.

- **Data storage and retrieval**: The data storage is implemented using SQLite. The APIs for data retrieval and data management can be accessed using HTTP GET methods. The APIs in turn access the SQLite database performing the desired actions.

B. *Software architecture of CCT*

The software architecture puts together all the modules of the IoT framework and is portrayed in the figure below.

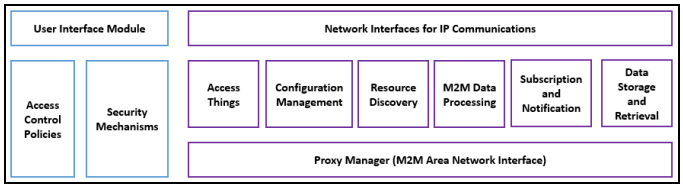

Fig. 3. High level software architecture for Connect and Control Things.

The prototype application contains a user interface. This provides the option for manually configuring the access control policies and selecting the security mechanisms. At the north interface, all interactions are based on standard IP communications while at the south interface, the interactions are part of an M2M area network

C. *Performance evaluation*

The prototype is installed in three Android powered devices – (i) Samsung Galaxy S2 running Android 2.3.4, (ii) Samsung Galaxy S3 running Android 4.3 and (iii) Nexus 7 running Android 5.1.1. This work presents some early results in terms of memory requirement, CPU usage, power consumption and time taken by the push notification component using GCM.

- **Memory requirements:** The generated APK file itself is less than 15 MB in size which establishes that the implementation of the IoT framework as CCT is

[1] https://code.google.com/p/androjena/

lightweight. It is observed that the configuration for each thing takes around 1KB memory. Thus an Android powered device can store configurations of thousands of things. Since the consumer smart devices are equipped with gigabytes of memory, data storage could store several hundred megabytes of processed data.

- **CPU usage:** The CPU usage is measured during several phases of operation of the application and the results are listed in Table I. It appears from the table that services for accessing things for metadata exchange, resource discovery and data storage & retrieval require very little CPU. Only the computing for M2M data processing registers relatively higher CPU load. This is due to the fact that the services other the M2M data processing are based on SenML and CoRE Link Format which are ultra-lightweight in nature. Creating or parsing JSON objects and arrays are done very easily. Also the SenML or CoRE Link payload of associated network operations are very less in size. Overall the CPU load is really less. On the other hand, the M2M data processing module includes a 3rd party library AndroJena and the implementation of the semantic reasoning engine is dependent on that. The complex nature of semantic computing attributes to the relatively higher CPU load. But that should not discourage consumers from using CCT. The CPU usage results point to huge improvement compared to our previous work [3].

TABLE I. CPU USAGE DURING DIFFERENT PHASES OF OPERATION

Android Device	Access Things	Resource Discovery	M2M Data Processing	Data Storage and Retrieval
Samsung Galaxy S2	1%	2%	14%	1%
Samsung Galaxy S3	1%	3%	18%	1%
Nexus 7	1%	1%	21%	1%

- **Power consumption**: The power consumption of CCT is measured using Power Tutor and portrayed in Table II. The measurements are average power consumption values taken over a complete life cycle of the application. A closer look into Power Tutor reveals that around 40% of the power is consumed by the M2M data processing module which could be accounted to the complex semantic computing. Another 34% power is consumed by the display hardware which can be lowered by reducing the brightness level. The overall performance of the mobile application is quite satisfactory. There is a lot of improvement in terms of power saving compared to our previous work [3]. We would further like to optimize the power consumption of CCT by following the guidelines mentioned in [16] [17].

TABLE II. POWER CONSUMPTION RESULTS FOR CCT

Android Device	Power consumption (mW)	
	Mobile Data	*Wi-Fi*
Samsung Galaxy S2	298	214
Samsung Galaxy S3	301	247
Nexus 7 (Wi-Fi only)	--	401

- **Time required by GCM operations:** Although GCM might appear to be a complex mechanism, it actually takes less than a second for the push notification. It is to be noted that the steps 1, 2 and 3 as seen in Figure 2 are done only once. Hence the notification depends only on steps 4 and 5. Although there are instances where GCM is used with BLE, in this work we have used GCM in conjunction with Wi-Fi.

V. USE CASE

IoT is increasingly being employed in smart home domain [9] [10]. We discuss home automation which perfectly fits as a use case of CCT. Utilization of CCT ensures that the home automation system need not depend on additional home gateway. This section describes the operational flows for to accomplish various home automation services.

A. Management of things in home automation

Both smart and legacy things may constitute home automation domains. The registration of thing descriptions to the configuration management module is outlined in Fig. 4.

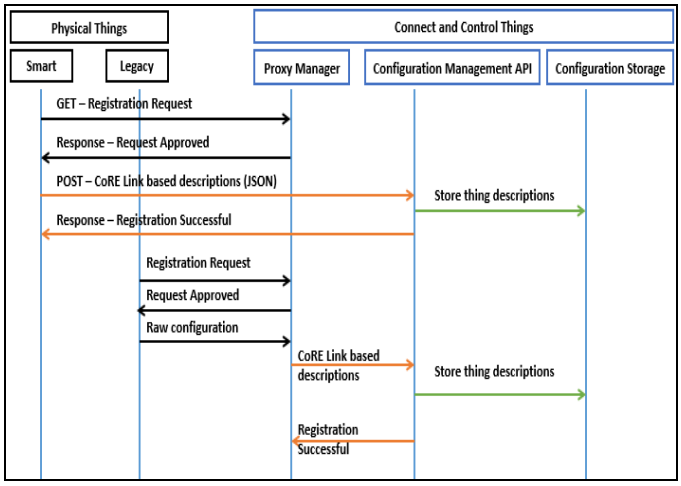

Fig. 4. Registration of thing descriptions to configuration management.

For smart things, it sends a GET request for registration which is approved by the proxy manager. Then the things

POST the descriptions directly to the configuration management API. But for legacy things, the associated proxy manager creates the necessary descriptions. Similar to this, the un-registration of things is achieved by sending a GET request for un-registration to the proxy manager. The corresponding configuration is deleted after the request is approved.

B. M2M data processing in home automation

The operational flows for M2M data processing module along with subscription and notification modules are combined and depicted in Fig. 5. The SenML metadata from a sensor is fed to the M2M data processing unit which generates actionable intelligence and is stored at the data storage. If a consumer device has subscribed to the occurrence of the event corresponding to the derived intelligence, a push notification is sent to the device. It can then use SenML extensions to send a command to the actuators.

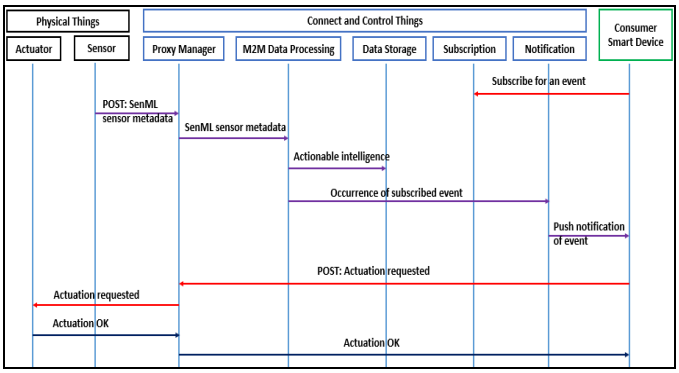

Fig. 5. Operational flow for M2M data processing and subscription and notification services.

VI. CONCLUSION

In a nutshell, the paper describes the functional requirements and architecture of a lightweight IoT framework. It is implemented as a mobile application (CCT) and is deployed to Android powered smart devices. The application can replace home gateways for home automation scenarios. The novel aspects of the framework and CCT are highlighted. The core functionalities of CCT are exposed to the consumer smart devices using RESTful web services. Interoperability is maintained among the modules and the overall framework is interoperable with similar IoT frameworks. The experimental results highlight the lightweight nature of design and implementation of the IoT framework as CCT as well as real time interaction among the components. This encourages adoption of CCT for home automation scenarios which is illustrated with operational flows. Finally we are working towards optimizing the M2M data processing module to further reduce the CPU usage and power consumption.

ACKNOWLEDGMENT

The authors would like thank Dr. Amelie Gyrard for the contribution on M3 framework. The paper is supported by French research project DataTweet (ANR-13-INFR-0008).

REFERENCES

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

[2] Datta, S.K.; Bonnet, C.; Nikaein, N., "An IoT gateway centric architecture to provide novel M2M services," Internet of Things (WF-IoT), 2014 IEEE World Forum on, pp.514,519, 6-8 March 2014.

[3] Datta, S.K.; Bonnet, C.; Nikaein, N., "CCT: Connect and Control Things: A novel mobile application to manage M2M devices and endpoints," Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2014 IEEE Ninth International Conference on, pp.1,6, 21-24 April 2014.

[4] Datta, S.K.; Bonnet, C., "Smart M2M Gateway Based Architecture for M2M Device and Endpoint Management," Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing(CPSCom), IEEE, pp.61,68, 1-3 Sept. 2014.

[5] Gyrard, A.; Datta, S.K.; Bonnet, C.; Boudaoud, K., "Standardizing generic cross-domain applications in Internet of Things," Globecom Workshops (GC Wkshps), 2014, pp.589,594, 8-12 Dec. 2014.

[6] Cirani, S.; Davoli, L.; Ferrari, G.; Leone, R.; Medagliani, P.; Picone, M.; Veltri, L., "A Scalable and Self-Configuring Architecture for Service Discovery in the Internet of Things," Internet of Things Journal, IEEE, vol.1, no.5, pp.508,521, Oct. 2014.

[7] Meirong Liu; Leppanen, T.; Harjula, E.; Zhonghong Ou; Ylianttila, M.; Ojala, T., "Distributed Resource Discovery in the Machine-to-Machine Applications," Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on, pp.411,412, 14-16 Oct. 2013.

[8] Riahi, A.; Challal, Y.; Natalizio, E.; Chtourou, Z.; Bouabdallah, A., "A Systemic Approach for IoT Security," Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on, pp.351,355, 20-23 May 2013.

[9] Yin Jie; Ji Yong Pei; Li Jun; Guo Yun; Xu Wei, "Smart Home System Based on IOT Technologies," Computational and Information Sciences (ICCIS), 2013 Fifth International Conference on, pp.1789,1791, 21-23 June 2013.

[10] Ming Wang; Guiqing Zhang; Chenghui Zhang; Jianbin Zhang; Chengdong Li, "An IoT-based appliance control system for smart homes," Intelligent Control and Information Processing (ICICIP), 2013 Fourth International Conference on, pp.744,747, 9-11 June 2013.

[11] 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, pp.202,207, 1-1 April 2012.

[12] C. Jennings, Z. Shelby and J. Arkko, "Media Types for Sensor Markup Language (SENML)", IETF draft work in progress.

[13] Datta, S.K.; Bonnet, C., "A lightweight framework for efficient M2M device management in oneM2M architecture," Recent Advances in Internet of Things (RIoT), 2015 International Conference on, pp.1,6, 7-9 April 2015.

[14] Z. Shelby, K. Hartke and C. Bormann, "The Constrained Application Protocol (CoAP)", IETF RFC 7252.

[15] TS-0001-oneM2M-Funtional-Architecture-V-2014-08 - http://www.onem2m.org/images/files/deliverables/TS-0001-oneM2M-Functional-Architecture-V-2014-08.pdf.

[16] Datta, S.K.; Bonnet, C.; Nikaein, N., "Minimizing energy expenditure in smart devices," Information & Communication Technologies (ICT), 2013 IEEE Conference on, pp.712,717, 11-12 April 2013.

[17] Datta, S.K.; Bonnet, C.; Nikaein, N., "Self-adaptive battery and context aware mobile application development," Wireless Communications and Mobile Computing Conference (IWCMC), 2014 International, pp.761,766, 4-8 Aug. 2014.

[18] Gyrard, A.; Datta, S.K.; Bonnet, C.; Boudaoud, K., " Cross-Domain Internet of Things Application Development: M3 Framework and Evaluation," 3rd International Conference on Future Internet of Things and Cloud, 24-26 Aug. 2015.