2015 International Conference on Recent Advances in Internet of Things (RIoT)
Singapore, 7-9 April 2015

A Lightweight Framework for Efficient M2M Device Management in oneM2M Architecture

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

***Abstract*—Recent years have witnessed an explosion in the number and types of physical devices connected to the Internet. This exponential growth in the volume of objects poses challenges in terms of managing the connected M2M devices. A unified approach for efficient management of the M2M devices while preserving scalability is necessary. This paper proposes an M2M device management framework that can be deployed in a cloud system, M2M gateway or even inside a mobile application. CoRE Link Format is used for lightweight description of smart M2M devices. The capabilities of CoRE Link are extended to describe legacy M2M devices as a part of Internet of Things ecosystem. Open Mobile Alliance Lightweight M2M (OMA LwM2M) Technical Specifications are used in the framework to provide M2M service enablement for end users. Self-management of the M2M device configurations is outlined. The capabilities of the framework are exposed using RESTful web services. oneM2M architecture for device management is described which integrates the proposed framework. Its software implementation is examined to be ultra-lightweight. Utilization of CoRE Link settles the heterogeneity of the managed devices and promotes interoperability. Finally the paper summarizes the contributions and concludes with future directions.**

Keywords—CoRE Link Format; M2M device; M2M endpoint; OMA LwM2M; oneM2M; Resource description.

I. INTRODUCTION

The Internet of Things (IoT) is extending the Internet connection to physical devices including sensors, actuators and RFID tags [16]. With the advancement in technologies, these devices are enabled with communication and computation capabilities. Several academic [10] [11] and industrial initiatives (e.g. IBM Smarter Planet Initiative[1]) are now trying to embed sensors and actuators to various walks of our daily lives utilizing next generation information and communication technologies. According to Cisco[2], around 7 Billion of such M2M devices are already connected to the Internet to provide advance services to the citizens and enterprises. As a result of that, many prototypes and services are being deployed for home automation, personal healthcare, traffic congestion management, waste management and more.

However the deployment of IoT based solutions in real world in a very large scale has not been that satisfying. One of the reasons for that is the lack of unified approach towards management of the huge volume of physical things. The task of M2M device management is itself very challenging due to – (i) heterogeneity of things in terms of communication technology (e.g. Bluetooth Smart, Zigbee, NFC, Modbus), measurement capabilities (e.g. temperature, luminosity, precipitation) and types of data generated (e.g. numerical, audio, video), (ii) high mobility, (iii) access control for end users, (iv) network topology being used, (v) self-configuration management, (vi) lack of uniform description, (vii) managing legacy devices and (viii) ease of adding and deleting M2M device description. Such challenges make the M2M device management a demanding process and a key task in an IoT ecosystem.

In this paper, we have gathered the requirements (described above) for automatic device management and accomplish that through a framework. It utilizes Open Mobile Alliance Lightweight Machine to Machine (OMA LwM2M) technical specification [14] for the service enablement for end users. A key part of efficient device management is the description of the devices. We have made use of CoRE Link Format[3] to represent the M2M devices and their endpoints using resource types and attributes. It provides an ultra-lightweight implementation which is suitable for the huge volume of devices and maintains the scalability in IoT ecosystems. We have also extended the capabilities of CoRE Link to describe legacy objects as well. The contributions of the paper are – (i) M2M device management framework that is capable of running from a cloud system, within a capillary network using M2M gateway and on a direct link to the mobile phone of an end user, (ii) CoRE Link Format based description of both smart and legacy devices, (iii) management of smart and legacy devices, (iv) utilization of OMA LwM2M based M2M service enablement for end users and (v) integration of the entire functionalities and components of the proposed framework into an oneM2M architecture[4]. A prototype of the architecture is implemented and evaluation results are provided. The framework also enables dynamic discovery of M2M device configurations and provides flexibility to the overall system.

The rest of the paper is organized as follows. Section II reviews the state-of-the-art and highlights their limitations. Section III focuses on the proposed M2M device management framework. Section IV describes the oneM2M architecture incorporating the framework with prototype implementation and evaluation results. Finally we conclude the paper in Section V.

[1] http://www.ibm.com/smarterplanet/us/en/?ca=v_smarterplanet
[2] http://share.cisco.com/internet-of-things.html
[3] https://tools.ietf.org/html/rfc6690
[4] http://www.onem2m.org/technical/candidate-release-august-2014

978-1-4799-8325-4/15/$31.00 ©2015 IEEE

II. STATE-OF-THE-ART

In this section we discuss the current state-of-the-art that has considered M2M device and network management. We also highlight their limitations.

A. M2M network management

Authors Pandey, Kim, Choi and Hong discuss the management of M2M networks [1]. They have identified the ongoing standardization efforts in ETSI, TIA, IPSO Alliance and OMA in relation to the network management, communication protocols and current status. Various unique characteristics of the M2M device and applications and their impacts on network management are presented. The important contribution is in identifying the M2M network management requirements which include fault management, automatic configuration management, remote software upgrade, mobility management, QoS monitoring, access control and security management. They have also reported M2M gateway based configuration management and registration of the M2M services. In addition to that, the M2M network management is in-charge of remotely (i) altering the state (online/offline) of the M2M devices, (ii) configuring specific parameters and (iii) upgrading the software.

M2M network management issues are further explored in [2] [3] which advocates for policy-based management. These papers define an M2M policy information model where users are policy makers and can create new policies, update the existing ones and delete if necessary. To ease the task, there is a policy editor interface. The created policy is translated in XML format and can be processed by a policy server. Policy based M2M network management is described as a solution where M2M devices with preconfigured policies can automatically respond to the events generated by the network.

B. M2M device management

The authors of [4] argue that a traditional management gateway can not handle dynamic entry and exit of smart objects in a network since unless there is software update or patch. This is due to the fact that current deployment scenarios do not require dynamic reconfiguration scenarios. Thus the authors have proposed a new approach – Management by Delegation Smart Object Aware System (MbDSAS). It utilizes discovery protocols to dynamically learn about the smart object behavior and they are stored as lists. Any top level management application can access those lists though a web service. The approach is integrated into three layer architecture and the paper also presents detailed prototype implementation along with performance evaluation.

OMA Device Management Protocol is presented and explained in [5]. Managing huge volume of M2M devices over LTE is a challenging task. The paper discussed SMS based wake-up mechanism which is remotely used to wake up M2M devices from sleep mode. This reduces the operating power consumption and increases the life time of the devices. The utilization of CoAP over HTTP in OMA-DM is preferred since the former consumes less power and has a better suited payload format. Efficient messaging formats (EXI, CoRE Link) are also outlined. In a nutshell, this paper gives an excellent overview of how M2M device management should benefit from OMA-DM Protocol.

A Device Connection Platform (DCP) has been discussed in [6] which is cloud-based, cost-effective solution to connect and manage M2M devices and associated applications. The architecture and deployment scenarios can handle both infrequent and frequent data exchange among the devices and the cloud. The paper also reports how the mentioned platform handles charging and accessing the devices via Internet, manages subscribed devices and enables access to several services.

C. Remote entity management

Remote entity management (REM) is explored in [7]. This paper highlights the possible M2M communication scenarios between the end users and M2M devices with the core network utilizing LTE. The REM for service capability in M2M networks is performed using a dynamic software update model. The main advantage is that it does not require the M2M devices to be rebooted.

In order to make M2M devices interact directly with the Internet, the devices should be configured with an entire IP-based stack. The paper [8] investigates integration of IP-based network management protocols into resource constrained M2M devices. Lightweight SNMP and NETCONF integrations have been described along with the key challenges which include message framing issue, session establishment and security. The main advantage of this work is that, M2M devices with such capabilities can be directly managed by end users over the Internet facilitating remote M2M device management.

Apart from the above, a survey conducted by Thoma, Braun, Magerkurth and Antonescu about management of things and services can be found at [9]. It highlights that self-management; interoperable solutions and management of things are not so widespread within the IoT community.

D. Limitations

The limitations of the state-of-the-art are given below.

- In case of [1], it provides a high level description of the requirements. There is no technical contribution in terms of how the physical devices and the network they belong to could be managed. However, the paper does not discuss any prototype implementation details or performance evaluation. Similarly [2] and [3] do not discuss any insight about the impact on the policy based management when the number of M2M devices scale to millions.
- The authors of [4] have not shown how the mentioned approach could fit into a standard IoT architecture like ETSI. In case of [6], it is not clear how each individual device is represented in the DCP and how the representation is updated when necessary
- The work [5] just contains high level overview. There is no discussion on architecture or prototype implementation or experimental evaluation to prove the effectiveness of M2M device management in LTE networks. Similarly [8] is centered on individual

devices employed in IoT. It does not consider describing devices and endpoints and how to manage them in an IoT ecosystem.

Apart from the points above, there is a lack of work on self-management of the M2M devices which is a key point in M2M communications. Also there is no way to manage legacy devices in an IoT ecosystem. Utilization of OMA based Lightweight M2M from IoT architecture is also not present. Our framework has been designed and developed to mitigate these limitations.

III. M2M Device Management Framework

In this section we describe M2M device and endpoint description, internal structure of the framework, its operational phases and the novel aspects of this work.

A. M2M device and endpoint description

Lightweight description of M2M devices is the key to implement an efficient and scalable device management framework. We utilize CoRE Link Format to describe the M2M devices and their endpoints. Our unique contribution is to extend the same description to legacy devices also. These are the objects that cannot respond to a GET request and communicates over a wired communication protocol like Modbus. To manage such legacy objects, they must be connected either to an intermediate gateway (IG) or a proxy which can interact with rest of the M2M elements over RESTful manner [12]. The IG or proxy is actually configured to represent the legacy devices. Apart from that, CoRE Link format settles the heterogeneity of devices in terms of measurement capabilities, generated data and communication technologies. Also it promotes interoperability among various components of the architecture.

We define a function set common to both the smart and legacy devices to represent them, their endpoints and other high level configurations as portrayed in Table I.

TABLE I. CoRE Link Based Function Set

Function Set	Root Path	Resource Type
Device	/d	wg.dev
Endpoint	/e	wg.endpoint
Configuration	/cf	wg.config

The function set "Device" represents the M2M device description and can be accessed by querying at the root path "/d". Table II portrays device resource description.

TABLE II. Device resource description

Type	Path	RT	IF
Location	/d/loc	ipso.loc.gps / ipso.loc.xy / ipso.loc.sem	p
Id	/d/id	wg.dev.id	rp
Name	/d/n	wg.dev.name	p
Model	/d/mdl	wg.dev.model	p
destination	/d/dst	wg.dev.destination	p
Lifetime	/d/lft	wg.dev.lifetime	p
proxy-out	/d/po	wg.dev.proxy-out	rp
proxy-in	/d/pi	wg.dev.proxy-in	rp

The "location" attribute is taken from IPSO Alliance Framework[5]. This attribute can be expressed using the GPS co-ordinates, semantic value or X-Y value with respect to a certain position and is used to keep track on the mobility of the devices. Among the rest, destination denotes the URI associated with the device. This URI is used to access the attached endpoints (e.g. sensors and actuators). The lifetime is used as a period to check whether the managed device is still present. In case of a legacy device, the proxy-out and proxy-in are used to generate the above description and to communicate with actuators and sensors respectively. The proxy attributes are our unique contribution and used to incorporate legacy objects into intelligent IoT systems [12]. It is to be noted that the proxy attributes are not needed for smart devices.

Similar to the M2M devices, the endpoints are described as portrayed in Table III. The "senml" attribute either contains the requested sensor metadata or pushed actuator control metadata. We have a uniform representation of metadata exchange for both sensor and actuator using Sensor Markup Language [12], [13].

TABLE III. Endpoint resource description

Type	Path	RT	IF
id	/e/id	wg.endpoint.id	rp
name	/e/n	wg.endpoint.name	p
device	/e/d	wg.endpoint.device	p
senml	/e/senml	wg.endpoint.senml	rp

The initial configuration combing the device and endpoint resources must be available as configuration resource description. During bootstrapping, a GET request is sent to retrieve this description which is portrayed below. This is used to expose the M2M device and endpoint configuration to the end users.

TABLE IV. Configuration resource description

Type	Path	RT	IF
Device	/cf/d	wg.dev	p
Endpoint	/cf/e	wg.endpoint	p

B. Device management framework

Figure 1 depicts the proposed M2M device management framework. It simplifies the management tasks by exposing the functionalities through RESTful web services. Thus complex policy based techniques are not required.

The perception layer contains the real M2M devices containing sensors, actuators or RFID tags as endpoints. The framework itself is composed of three layers of web services namely service enablement layer, configuration storage layer and proxy layer.

- **Proxy Layer –** This is a novel aspect of the framework which allows management of non-smart or legacy

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

M2M devices in IoT ecosystems. Current standardization efforts do not consider such scenarios but inclusion of legacy devices into IoT ecosystems is crucial. The proxy layer is composed of two RESTful web services – proxy-in and proxy-out to manage sensors and actuators respectively. The proxy layer creates the CoRE Link based configurations and is responsible for registering and un-registering legacy devices. The proxies are dependent on the communication protocol used by the legacy devices.

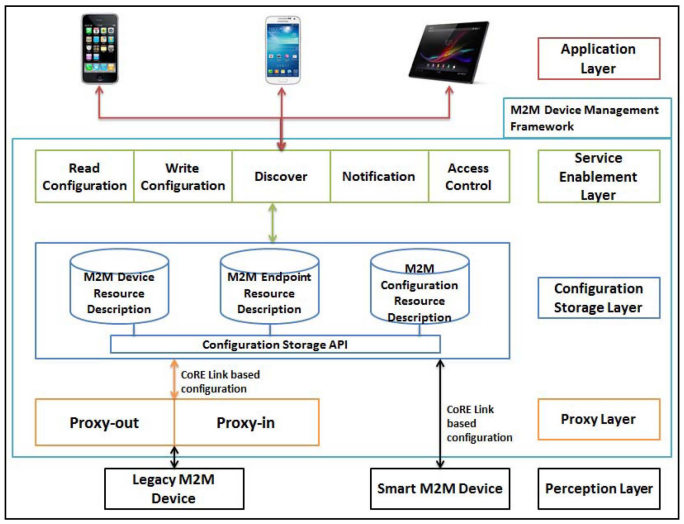

Fig. 1. M2M device management framework.

- **Configuration Storage Layer** – This contains a web service called "Configuration Storage API". The smart devices directly connect to this API during the bootstrap phase and the API extracts the resource descriptions from the devices or (proxies in case of legacy devices). The layer houses a database and stores the device, endpoint and configuration resources in separate tables. The API translates the CoRE Link based descriptions to appropriate storage format. This layer also keeps track of the configuration "lifetime" attribute. During that period, if it does not receive an announcement that the device is still present or configuration update, it will delete that device configuration.
- **Service Enablement Layer** – It is composed of several web services which allow the end users to (i) read, write & update configurations, (ii) enable device discovery, (iii) receive notification and (iv) implement proper access control[6]. These capabilities correspond to OMA LwM2M technical specification [14] and allow remote management of M2M devices from mobile devices of end users. The functionalities of these web services are explained below.

[6] Discussion on access control is out of scope of the paper.

C. *Deployment scenarios*

The overall framework is highly generic in nature which allows it to be efficiently deployed in M2M gateways, cloud based systems or even inside a mobile application depending on use case and requirements. For a large enterprise consisting of dozens of smart objects using various different technologies to communicate, M2M gateway can be employed for efficient device management. Using the proxy layer, the gateway can include legacy devices along with smart devices. Cloud based management is required when we consider the M2M devices deployed in a smart city. The proposed framework will offer a scalable solution to manage the thousands of devices. For a smart home with limited number of M2M devices can utilize a smartphone or tablet to manage them. The framework in that case will be implemented inside a mobile application which can interact with objects using a personal area network. For every scenario mentioned above, the devices must register themselves to the framework. Following is the mechanism for that.

D. *M2M device registration phase*

This phase is used to provision initial configuration from the M2M device to the configuration storage API. We assume that the device is already configured with bootstrap information which corresponds to "Factory Bootstrap" procedure of OMA LwM2M technical specification [14]. The operational sequence for both smart and legacy devices registration is presented in Figure 2.

For a smart device, it is quite straight forward. It connects to the web service of configuration storage API which approves the request. Then the device pushes the CoRE Link based description. The interactions take place in a RESTful manner. In case of a legacy device, it connects to the respective proxy at first which is treated as the registration request. Then it uploads the raw configuration and the proxy translates that into CoRE Link description. Thus the proxy must be configured with the additional intelligence during its bootstrap. This step is followed by registration of end user applications running on smartphones or tablet to the framework. This is handled by the access control web service at service enablement layer. The access control mechanisms also determine which user has access to which M2M devices and provides a secure access to those devices.

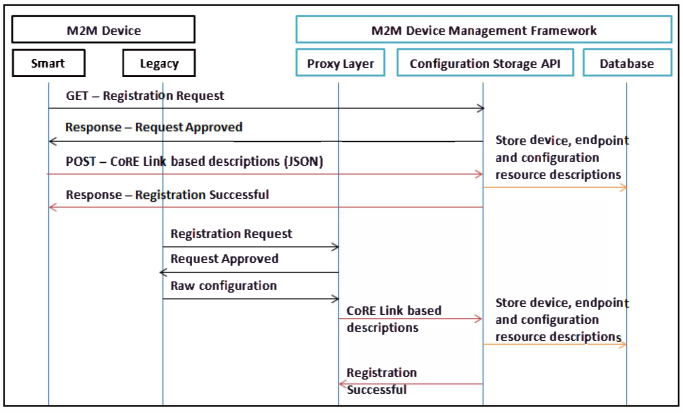

Fig. 2. M2M device registration phase.

E. *Service enablement phase*

Authorized end users can access the M2M device configurations using the other web services of service enablement layer. The end users can discover the configurations of the M2M devices and endpoints by sending a get request using device and/or endpoint ID. The web service internally queries the database and replies with the full configuration if a match is found. To update any description, "Write Configuration" is used. In this case, the end users send a PUT command which is initially routed to M2M configuration resource. This further checks whether the requested device is still attached (using lifetime) and if so, whether the requested description is allowed to be updated. The notification web service allows pushing information when there is a change in configuration in the M2M devices. If the client enables these services, push notifications are sent if (i) configuration is updated, (ii) new M2M device registers for a user and (iii) any device un-registers. This enables the users to be aware of the state of the devices.

F. *Self-management of M2M devices*

Self-management of M2M devices is an important requirement in IoT based services. This is necessary as many of the devices may be mobile. Thus a lifetime attribute is added to the M2M device configuration (based on the OMA LwM2M specifications). The value of lifetime depends on actual use case and can be updated by the end users. Each device must announce itself or update its configuration to the framework once during the lifetime otherwise the device is removed from the configuration storage layer. It corresponds to a device being un-registered and a notification is sent to the corresponding end user.

G. *Un-registration phase*

The devices can send a un-registration request to the framework using "DELETE" configuration. In that case the, configuration storage API deletes the descriptions of the M2M devices identified by the device ID and may notify the corresponding end user. Also, if a device does not announce itself during the 'lifetime', it is automatically unregistered.

IV. Integrating M2M Device Management Framework in oneM2M Architecture

The entire framework and the operational phases mentioned above are fully integrated into an architecture based on recently released oneM2M recommendations [15]. The functional architecture components are presented in Figure 3 and broadly categorized into field and infrastructure domains. The elements of each domain are composed of two entities. An Application Entity (AE) contains the application logic for the end-to-end M2M solutions e.g. application for automated driving, fitness monitoring. Each AE must have a unique identity. On the other hand, a Common Service Entity (CSE) represents a set of common service functions of the M2M ecosystem. The M2M device management framework is included in the CSE. The following subsections further illustrate the components.

A. Application Service Node

An application service node (ASN) contains at least one AE and a CSE. In this case, the ASN-AE and ASN-CSE are embedded into the mobile application running in the smartphones or tablets. There are two cases two consider here. Firstly, the mobile application provides a user interface through which the end users can interact with the framework located outside the application. In that case the ASN-AE just contains a user interface (UI) and the ASN-CSE implements the modules for parsing CoRE Link description to show it in UI. Secondly, the mobile application includes the entire framework in ASN-CSE which will include the proxy and configuration storage layers.

A prototype Android application called "Connect and Control Things" (CCT) is developed considering both the cases. Porting the framework in Android is very easy as it supports CoRE Link descriptions through JSON and database using SQLite. Android also supports creation of capillary network using Bluetooth which allows Bluetooth based legacy devices to communicate with the mobile application. The resulting application is lightweight in terms of memory and CPU requirements.

B. Middle Node

A middle node (MN) contains only CSE and not AE. It communicates with an infrastructure node (IN) and an ASN. The M2M gateway corresponds to the MN. The MN-CSE implements the proposed framework. In this case, the web services are categorized as north and south interfaces. The north interface corresponds to the service enablement layer and facilitates interaction with ASN (end user mobile application). The notifications are sent using Google Cloud Messaging (GCM) which is a push notification framework for Android applications [12]. The south interface included the proxy layer and configuration storage layer and facilitates interaction with physical devices.

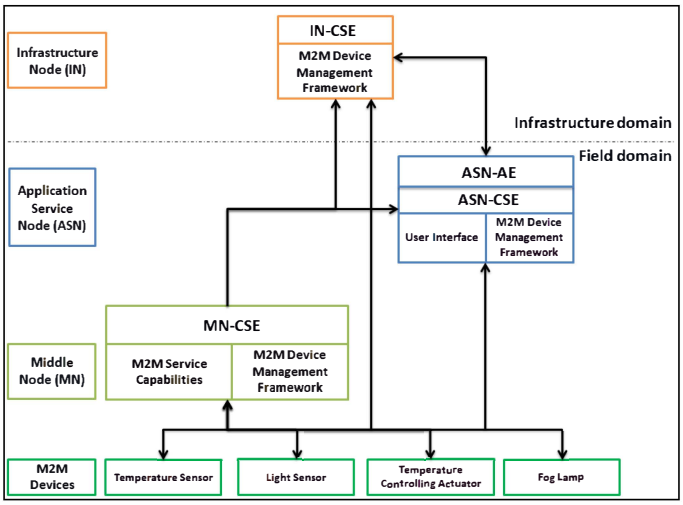

Fig. 3. M2M device management framework integrated into oneM2M architecture.

In the prototype implementation, we considered a legacy sensor device connected to the proxy over Modbus protocol. The proxy is pre-configured with an API which translates the

raw sensor configuration into CoRE Link. A smart device is also taken into account which follows the mentioned registration process. The MN-CSE is running on a Raspberry Pi currently.

C. Infrastructure Node

An infrastructure node (IN) provides an M2M service in the infrastructure domain. IN contains a CSE and none or more AE. This node interacts with one or more MNs and/or one or more ASNs. IN basically corresponds to a cloud system. The IN-CSE in this case contains the proposed framework. It can obtain the M2M device configurations from smart devices capable of communicating over Wi-Fi, 3G or LTE. Otherwise, it can also gather the configurations from various M2M gateways deployed around a smart city. In this case the ASN connects to the IN-CSE to read, write, discover or receive notifications. This implementation is ongoing.

D. *Evaluation*

The size of configuration of each M2M device including the endpoint is less than 1KB. Therefore smartphones, tablets and even M2M gateways with gigabytes of internal memory will be able to support and manage numerous such devices. The software implementations of the framework for M2M gateway and Android application require couple of megabytes of memory. It further assures the ultra-lightweight nature of the proposed mechanisms.

V. CONCLUSION

In a nutshell, the paper identifies the limitations of current literature on M2M device management. Germinating from the limitations, we have proposed a framework to efficiently manage both smart and legacy device. This is an important feature for IoT ecosystems. The core configuration storage is exposed to end users and physical devices using appropriate web services. The end users can perform a wide range of operations including read & write configurations to authorized devices; discover configurations and receive notifications. Another novel aspect is the self-management of configurations using a lifetime attribute. It takes care of the devices with high mobility. The overall framework is integrated into oneM2M architecture which is a distinct contribution of the paper over current literature. Use of CoRE Link ensures interoperability among the architecture elements. It is shown that framework can be operational from cloud systems, M2M gateway or a mobile application. This is another novel aspect of this work. Our evaluation shows that the software implementation has limited memory footprint and the device descriptions are ultra-lightweight. This takes care of the scalability issue if number of managed devices grows exponentially. Our future efforts are focused on implementation of the mentioned IN-CSE and access control to devices.

ACKNOWLEDGMENT

The work is sponsored by French research project Data Tweet.

REFERENCES

[1] Pandey, S.; Mi-Jung Choi; Myung-Sup Kim; Hong, J.W., "Towards management of machine to machine networks," Network Operations and Management Symposium (APNOMS), 2011 13th Asia-Pacific, vol., no., pp.1,7, 21-23 Sept. 2011.

[2] Kamal, R.; Siddiqui, M.S.; Haw Rim; Choong-seon Hong, "A policy based management framework for machine to machine networks and services," Network Operations and Management Symposium (APNOMS), 2011 13th Asia-Pacific , vol., no., pp.1,4, 21-23 Sept. 2011.

[3] Siddiqui, M.S.; Ahmed, S.H., "Policy-based network management in a machine-to-machine (M2M) network," Multitopic Conference (INMIC), 2012 15th International , vol., no., pp.387,393, 13-15 Dec. 2012.

[4] Marotta, M.A; Carbone, F.J.; Cardoso de Santanna, J.J.; Rockenbach Tarouco, L.M., "Through the Internet of Things -- A Management by Delegation Smart Object Aware System (MbDSAS)," Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual , vol., no., pp.732,741, 22-26 July 2013.

[5] Gligoric, N.; Krco, S.; Drajic, D.; Jokić, S.; Jakovljevic, B., "M2M device management in LTE networks," Telecommunications Forum (TELFOR), 2011 19th , vol., no., pp.414,417, 22-24 Nov. 2011.

[6] Cackovic, V.; Popovic, Z., "Device Connection Platform for M2M communication," *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on* , vol., no., pp.1,7, 11-13 Sept. 2012.

[7] Yao-Chung Chang; Ting-Yun Chi; Wei-Cheng Wang; Sy-Yen Kuo, "Dynamic software update model for remote entity management of machine-to-machine service capability," Communications, IET , vol.7, no.1, pp.32,39, Jan. 4 2013.

[8] Sehgal, A.; Perelman, V.; Kuryla, S.; Schonwalder, J., "Management of resource constrained devices in the internet of things," Communications Magazine, IEEE , vol.50, no.12, pp.144,149, December 2012.

[9] Thoma, M.; Braun, T.; Magerkurth, C.; Antonescu, A-F., "Managing things and services with semantics: A survey," Network Operations and Management Symposium (NOMS), 2014 IEEE , vol., no., pp.1,5, 5-9 May 2014.

[10] Sanchez, L.; Gutierrez, V.; Galache, J.A.; Sotres, P.; Santana, J.R.; Casanueva, J.; Munoz, L., "SmartSantander: Experimentation and service provision in the smart city," Wireless Personal Multimedia Communications (WPMC), 2013 16th International Symposium on, pp.1-6, 24-27 June 2013.

[11] Yongmin Zhang, Interpretation of Smart Planet and Smart City [J]. CHINA INFORMATION TIMES, 2010(10):38-41.

[12] 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 , vol., no., pp.514,519, 6-8 March 2014.

[13] 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 , vol., no., pp.1,6, 21-24 April 2014.

[14] http://technical.openmobilealliance.org/Technical/technical-information/release-program/current-releases/oma-lightweightm2m-v1-0

[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] Luigi Atzori, Antonio Iera, Giacomo Morabito, The Internet of Things: A survey, Computer Networks, Volume 54, Issue 15, 28 October 2010, Pages 2787-2805.