Managing the Internet of Things

Mahmoud Elkhodr, Seyed Shahrestani, and Hon Cheung
School of Computing, Engineering and Mathematics
University of Western Sydney
Sydney, Australia

***Abstract*—The Internet of Things (IoT) offers many interesting solutions to problems encountered in a range of application areas. The potentially huge number and the diversity of things that may be part of such an infrastructure can pose prominent issues for their systemic management. This paper explores such issues and proposes a management platform that partially addresses them. More specifically, our proposed IoT Management Platform (IoT-MP) takes into account the fact that things, in general, have limited power, computation, and communication resources available to them. The proposed platform supports fundamental management functions, including those needed for proper operation, monitoring and communications. After introducing the architecture and the major components of the platform, we show the advantages of our proposed approach through some simulation studies.**

Index Terms—Internet of Things, Management, Platform, Architecture.

I. Introduction

The Internet of Things (IoT) is an evolved concept of communications in which connectivity and, possibly, intelligence are added to just about every object found in a real world environment. From a technical point of view, the IoT is about giving an object the computing and communications capabilities to connect to the Internet [1]. All current predictions of the number of connected things to the Internet are enormous. For example, Cisco estimates that 50 billion of things will be connected to the Internet in 2020 [2]. Significantly, the vast number of connected things to the communication networks, such as smart appliances, smart automobiles, sensors, actuators, and others raise some challenges such as those reported in [3]. Indeed, more IP addresses are required for things [4]. Communication networks such as wireless and mobile networks must be capable of handling the massive increases in the Internet traffic generated by things [5].

IoT faces many significant issues. The potentially massive interconnection of things and their diversity create management challenges. This is particularly the case in the areas of data storage and management, operations, monitoring, maintenance, and performance, among others [6]. For instance, from operation and control points of view, it is essential to know the status of things, whether they are running, listening, or down. The capabilities of turning things on and off, disconnecting things from specific networks and connecting things to other networks are examples of the requirements needed to control the operation of things. Other issues relate to the maintenance of thing [7]. There is a need to detect the failure of things, particularly when the IoT involves a large interconnection of things depending on one another for communications. The detection of failure is also vital for things that may be deployed when availability is of significant importance, for example deployment in remote locations as essential parts for some emergency applications [8].

There are also the more traditional security challenges that need to be addressed [9]. These may be related to authorization, authentication, and access control. Things should be securely connected to their designated networks and securely accessed. Data generated or gathered by things need to be collected, analyzed, stored, dispatched, and presented in a secure manner. Moreover, there are specific security issues associated with thing-to-thing communications. For instance, if things are to be accessed by applications or software independent from the human users, then several security measures need to be enforced [10]. These measures should ensure that things are not leaking information and disclosing private information to unauthorized things. They should ensure that things are not used malciously to spy or harm the network in any other manner. With regards to privacy, things have their users and owners. Thus protecting the privacy of the user needs to be considered as well [11].

There is also another significant challenge that needs to be addressed. It relates to the nature and capabilities of things [12]. As it stands, the resources of things, their memory, processing power, power supply, and the like are very limited. This poses serious challenges for the deployment of many traditional management and security approaches. For instance, it is hard to achieve security on tiny devices compared to traditional computation devices [13]. Additionally, things or groups of things are often deployed in remote areas or in areas where accessibility is an issue; which makes otherwise trivial task of changing the batteries of things a difficult one [14].

The unique characteristics of the IoT, demand either new management methods or approaching the prevailing management systems differently. There is a need to manage the unprecedented number of things connected to the Internet that generate a large amount of traffic, particularly the devices with low power capabilities. To address these concerns, this work introduces an Internet of Things Management Platform (IoT-MP). To our knowledge, this work is amongst the first studies that address management of things as part of the IoT.

The remainder of this paper is organized as follows: Section II introduces the IoT-MP. In Section III, the IoT-MP basic components are described. The communication architecture of the IoT-MP is given in Section IV. Some management functionalities provided by the proposed platform are demonstrated in this Section as well. The simulation and implementation works are reported in Section V. Conclusion remarks, and future works are presented in Section VI.

978-1-5090-0214-6/15 $31.00 © 2015 IEEE
DOI 10.1109/DSDIS.2015.12

II. MANAGING THINGS IN THE IoT

The aim of the proposed IoT-MP is to provide a platform that facilitates and supports the management and communications of things on a communications network. Things can be any physical or virtual objects that have the communication capabilities of sending and/or receiving data to/from the communications network. A communications network is any medium that transfers data collected by things to other things or users. It is also a medium for transferring instructions from things to things such as remote actuation. Things range from simple tiny devices with sensing or actuation capabilities to sophisticated, intelligent devices also referred to as smart objects or devices. Application examples of the IoT can be found within the supply chain and healthcare industries. In the supply chain industry, things can be attached to physical objects such as mail orders, shipment containers and a variety of other goods. These things can provide information about how goods are handled during transport. This information helps in the real-time monitoring and tracking of goods as well. Also, installing things on transport vehicles helps in collision avoidance, traffic jam control, and better navigation. On the other hand, the IoT provides new opportunities for the healthcare industry. For instance, body sensors devices placed on a patient can be used to provide real-time information on the patient's status. Tracking of medical assets, patient-flow monitoring, and controlling accesses to restricted areas are IoT application examples in the healthcare sector as well.

The state of the art application in the IoT is a sophisticated smart system consisting of devices that sense data and identify users, animals, other things and events in their environment. This system processes these data and communicates the data with another system. Also, it converts the data into instructions that feedback through the communication networks to other devices with actuating capabilities that in turns actuate other devices, eliminating many human interventions.

To this end, the proposed IoT-MP is a platform offering human users with management functions, supervision and control over things. Things supported by the IoT-MP are referred to as Managed Things (MTs). The platform supports management of services such as sensing, actuating, monitoring, and managing devices locally and remotely over the communications network. Management of other services such as data storage and notifications are also supported by the platform. Thus, the IoT-MP does not only support the communications between thing-to-things and things-to-person, but also allows the management and control over things and their communications. A high-level view of the IoT-MP is given in Fig 1.

III. STRUCTURE OF THE MANAGEMENT PLATFORM

The basic organizational model of the IoT-MP is a simple two-tier model. It consists of an Agent that may be residing on the Managed Thing (MT), or on its local area network on one.

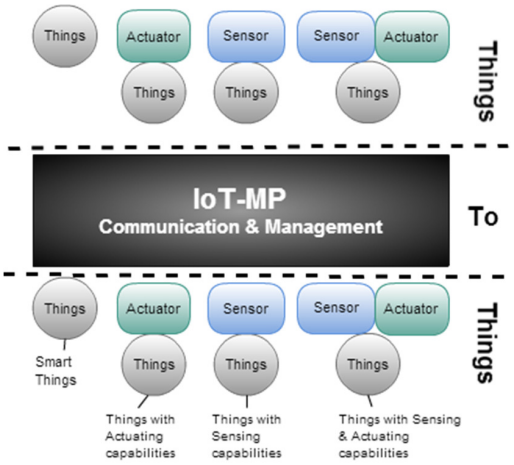


Fig. 1. Things to Things communication using the IoT-MP.

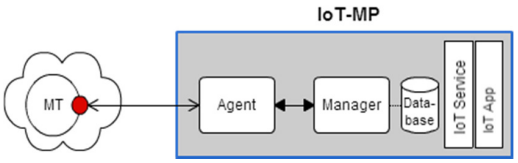


Fig. 2. Block diagram of the IoT-MP

hand. On other hand, it consists of a Manager, which resides somewhere on the communication network and manages the Managed Things (MTs). Multiple Agents can interact with a Manager. Managers can interact with other Managers as well. A Manager can act as a proxy Agent for other Managers, and a Manager can act as a Manager of Managers. Therefore, the model architecture can vary from a simple two-tier model to a more complex model of a hierarchal and distributed structure consisting of multiple Agents and Managers. This is to cater to the diversity and heterogeneity of devices and networks envisioned in the IoT. IoT applications and services are built on top of the Manager enabling communications between things via their respective Managers. The organizational model of the IoT-MP is shown in Fig 2. The representation of a MT in the IoT-MP is described in more details in the next section. The database shown in Fig 2 is discussed in Section III.

A. Managed Things

Managed Things (MTs) are virtual representations of things in the IoT-MP. MTs have attributes that describe them, referred to as management attributes. Also, MTs have behavioral attributes that hold the data they sense or the information related to actuation events. Therefore, MTs can be accessed and viewed through their attributes. This schema allows Agents, Managers, users, IoT services and applications to view and access the MT's attributes. Access policies on the attributes can also be defined by the Manager. These policies can be used to control access to data.

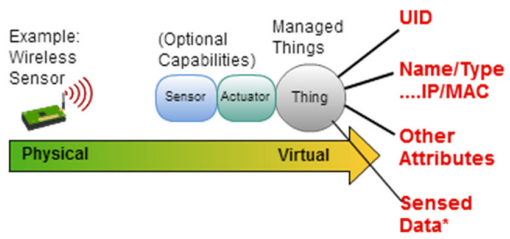


Fig 3- Things Attributes

The access policies and privileges to these attributes are controlled by superior Agents or Managers. Consequently, this structure provides human administrators/users with a way to manage MTs on one hand, and the application of disclosure policies on these attributes on the other side. Thus, controlling access to the data generated or collected by things. Some of the attributes relating to the description of MTs are listed below and shown in Fig 3:

- Device Identifier: This is a unique ID (UID) that identifies an MT from another.
- Name: This describes the name of the MT or its functionality e.g. Sensor
- Syntax: used to model the MT

These parameters along with other behavioral parameters are further discussed in Subsection D below.

B. Agents

In the IoT-MP, each MT speaks with an Agent. The Agent is part of the IoT-MP hierarchal structure. The Agent interacts with a Manager. It handles incoming information from an MT. The Agent collects information from an MT locally and makes this information available to the Manager. An Agent is also responsible for passing the requests coming from the Manager to the MT. Also, an Agent can collect management information about its environment, sends notifications to a Manager, and sends or retrieves information to or from the Manager's database. In the case where an MT does not possess a location-aware capability, its Agent can be assigned the responsibility of communicating the MT's location to the Manager. In some cases, an Agent can act as a proxy for other Agents or Managers as well. An Agent has the following responsibilities:

- Listening to updates from an MT.
- Requesting updates from an MT.
- Receiving requests from and sending responses to a Manager
- Receiving actuation instructions from a Manager and passing them to the MT.
- Reporting MT statuses to a Manager.
- Reporting locations of the MT to the Manager.

C. Managers

A Manager issues requests to an Agent, which functions as the equivalent of a server. The Manager is an application that performs the operational roles of generating/forwarding requests to retrieve information from an MT. It can also execute actuation requests on particular MTs which have actuation capabilities. The Manager receives notification reports on MTs

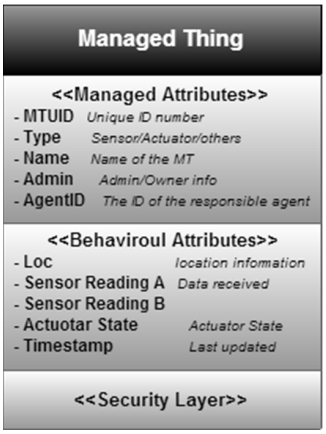


Fig. 4. Managed Things Attributes on the Database

generated by their Agents as well. Additionally, it issues requests for management's operations on behalf of an administrator or IoT application and receives management notifications from Agents. The Manager accesses an MT transparently as if the MT is just one hop away. It relies on an Agent in doing this. A Manager maintains a database of MTs, their Agents and the data they generate. Significantly, a Manager executes applications that monitor and control MTs. The Manager supports access control, security, privacy, interoperability among other management services. To handle and store the information sent/received from MTs via Agents, the Manager uses a Management Database.

The "Thing-to-Agent" and "Agent-to-Manager" arrangement provide distributed management capability over the IoT-MP, in which a Manager or Agent can act as a proxy for other Agents or Managers.

D. Management Database

The attributes of things are stored in the database that is located at the Manager. Therefore, each MT is represented on the Manager's database. An example of how MTs are represented on the Database is given in Figure 4. The Managed attributes module describes the MT's type, name and the related management attributes. The Behavioral attributes module stores the data received by the Manager from an MT via its Agent.

IV. COMMUNICATION ARCHITECTURE

The communication architecture of the IoT-MP is divided into two sub-models: the "MT to Agent" sub-model and the "Agent to Manager" sub-model.

A. The MT to Agent Sub-Model

An Agent is the first point of contact between an MT and the IoT-MP. An Agent acts as a bridge allowing information to flow from an MT to the IoT-MP. An Agent's role is to provide communications capabilities allowing the exchange of information between an MT and the IoT-MP. An Agent consists mainly of two components: a Communication API and Information API. These components are directly associated

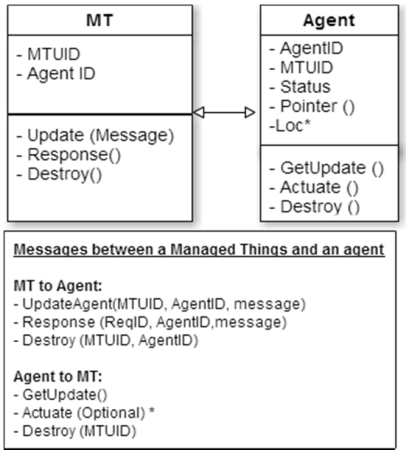


Figure 5- Information API

with the MT and usually reside on the MT. The Communication API includes protocol and network capability layers. It establishes the communication protocol that is used for the transmission of data from the MT to the IoT-MP and vice versa. For instance, for a particular MT, the Agent's Communication API could define ZigBee as the communication protocol. In another application, for example, the Communication API can use Bluetooth 4.0 as the communication protocol. Ultimately, this architecture enables an interoperability of thing to things communication despite the use of different communication protocols. On the other hand, the Information API, as shown in Fig 5, defines the types of messages that can be exchanged between an MT and an Agent. It also includes a definition of the message format and error handling.

1) Messages between a Managed Thing and an Agent:

The followings are some of the messages exchanged between an MT and its Agent:

- *Update(MTUID, AgentID, Message)*: this message is sent from an MT to its Agent. It has three arguments: the MTUID stands for the Managed Thing Unique Identifier ID. This ID is used by the Agent to identify uniquely an MT. The second argument, AgentID, is used to allow an MT to communicate with its Agent. The last argument, Message, contains the content of the message.
- *GetUpdate()*: This message is sent from a Manager and is used to request an update from the Agent of an MT.
- *Response(ReqID, AgentID, Message)*: this is a response message from an Agent to the Manager's message GetUpdate. This response message is only triggered by the GetUpdate message from the Manager. The Manager uses ReqID for matching requests to responses.
- *Actuate()*: This message is sent from a Manager to an Agent, triggering an actuation event on the MT.

The *Update* message format is shown in Fig 6. It includes the MTUID, Error Status, and Message as entries. The *Actuate* and *Ack* message formats are given in Fig 7; while *GetUpdate* and *Response* message formats are given in Fig.8. The *GetUpdate* message argument ReqID is a unique ID used to identify requests for matching them later against responses. In all messages, AgentID is used to determine Agents to Managers. The message *Actuate* is an optional message. It is only supported by MTs which have actuation capabilities. It consists of two messages: *Actuate* and *Ack*. The entry Status reports the current actuator status in the *Ack* confirmation message to the Agent. For example, suppose the actuator on the MT can turn an object ON or OFF. Therefore, the ON or OFF message will be the status reported back to the Agent in the *ACK* message in this example. Significantly, the Manager accesses the MTs transparently via their Agents.

B. The Agent to Manager Sub-model

The messages exchanged between an Agent and Manager are divided into two categories. The first relates to the management of the communications between an Agent and its Managers. It is related to matters such as managing the association between agents and a manager. The second category of messages relates to information retrieval/update from/to a Managed Thing.

1) Management Messages

The following is a list of management messages exchanged between a Manager and an Agent along with the description of each one:

- *GetMTStatus(MTUID, [AgentID])*
- *GetUpdate(MTUID, [AgentID])*
- *Actuate (MTUID, [AgentID])*
- *GetDeviceMode ([SensorMode], ActuatorMode , GeneralMode, [AgentID])*
- *GetLoc (MTUID, [AgentID])*
- *SetLoc (MTUID)*

The SensorMode and ActuatorMode in the message *GetDeviceMode* are optional as they depend on whether a Managed Thing possess sensing and actuating capabilities or not. The *GetLoc* message is a request for the last known location of an MT. An MT, which does not possess location awareness capabilities, relies on its Agent to provide its location. SetLoc is restricted to an MT with a fixed physical location or a location that could change within fixed limited boundaries e.g. the location of an item on a shelf in a warehouse. The *SetLoc* and *GetLoc* messages are part of the Location Privacy Management Module (LPMM). The LPMM is used to control the privacy of things on the IoT-MP. The LPMM is out of the scope of this paper and will be discussed in future works.

V. SIMULATION STUDIES

An IoT Smart Home scenario which implements the proposed IoT-MP has been simulated using the Network Simulation software version 2.0 (NS2). The aim of this simulation works is to validate and test the applicability of the proposed IoT-MP in low rate wireless personal area networks, such as on a ZigBee network. The implemented scenario consists of several devices that communicate over ZigBee.

Messages Format

MTUID	Loc(optional)* *GetLoc(),SetLoc()*	Agent Destination (Agent ID)	Error Status
<<Message>>			

**Loc refer to location. This is optional and depends on whether the MT has location awarness capabilities*

Error Status

Error Index	Description
0	No errors
1	UpdateAgent error
2	Response error
3	Location Unavailable
4	General error

Fig 6. Update Message Format

Messages Format: Actuate()

MTUID	ReqID	Agent ID (Optional)
Message (Actuate)		

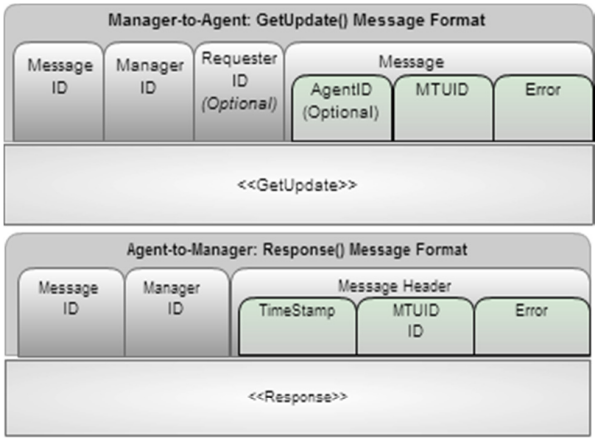
Messages Format: Ack()

MTUID	ReqID	Error Status
Status		

Error Status

Error Index	Description
0	No errors
1	Function not supoorted
2	Actuate error
3	success
3	Unauthorised
4	General error

Fig. 7. Actuate and Ack Messages' Format



Field	Description
Message ID	An ID used to match the *GetUpdate()* requests to the *Response()* message
Manager ID	An ID used to identify managers by agents
Requester ID	An optional ID that identifies the entities requesting the update e.g. another MT
Agent ID	Used to uniquely identify agents repsonbile of objects
MTUID	An ID that uniquely identifies managed Things
Error Index	an index that desribes the nature of the error

Fig. 8. GetUpdate() and Response() Messages

It is based on the scenario described in the ZigBee IP technical specification document [15]. The network setup of this scenario consists of several ZigBee devices acting as smart appliances, smart plugs or simply temperature sensors. These devices communicate sensory data to a ZigBee router device which in turn routes the data to a ZigBee coordinator device. The router device, as described in the ZigBee IP specification example, can be any device that has an additional function of routing the data onto the network. On the other hand, the ZigBee coordinator can be a programmable communicating thermostat with advanced support for an in-home display [15].

Consequently, the Manager and Agent described in the proposed IoT-MP architecture has been implemented on the coordinator device and router devices respectively. On the other hand, ZigBee devices are implemented as Managed Things (MTs). Therefore, using the IoT-MP platform, the ZigBee devices (MTs) periodically send data updates via their respective Agents to the Manager, which stores the received information in the Manager's database. This distributed architecture allows a user to monitor and control the operation of things (e.g. the appliances) remotely over the Internet. It provides users with the management capabilities of the ZigBee network. Figure 9 shows the simulation scenario, and Table 1 displays the ZigBee configuration parameters. From Fig. 9, Node 3 and 4 represent the sensors. Node 1 and 2 act as Agents, while Node 0 acts as the Manager. This node has a database as well. Node 5 and 6 are management applications in the form of requesters. They request information on Node 3 and 4 including their status.

Communications on the network are as follows: Nodes 3 and 4 periodically transmit specific data that are the SensorID, temperature, and location using ZigBee respectively to nodes 1 and 2 within their hearing range. Next, the Agent nodes process the received messages by adding their own data, such as the Agent ID and a Timestamp and forward the messages to Node 0 (the Manager) also over ZigBee. The Manager stores the information received in a local database. At a given time, Nodes 5 and 6 (requesters) request Node 3's and 4's locations and status updates from the Manager. The Manager can respond by retrieving, from the database, the last received updates from the sensors (Node 3 or 4). Alternatively, a real-time status update can be requested from a sensor as shown in Figure 10.

Alerts can also be configured to be triggered based on particular events, e.g. an alert can be created once the temperature of Node 3 (the sensor) drops below a certain threshold. Figure 11 shows the application traffic received in packet per seconds, and sensors' battery consumed energy. It indicates that the battery consumption remained at an acceptable level of less than 0.015 Joules.

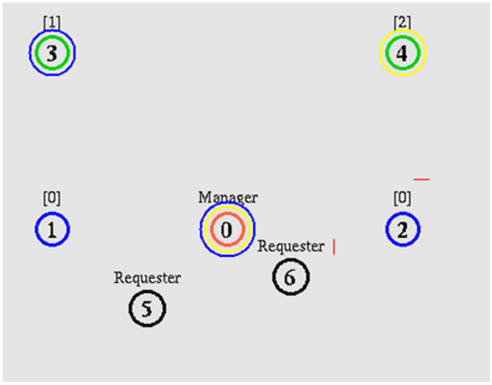


Figure 9: Sensor Node 4 is sending status update to Manager via Agent Node 2 (Traffic in Red Line)

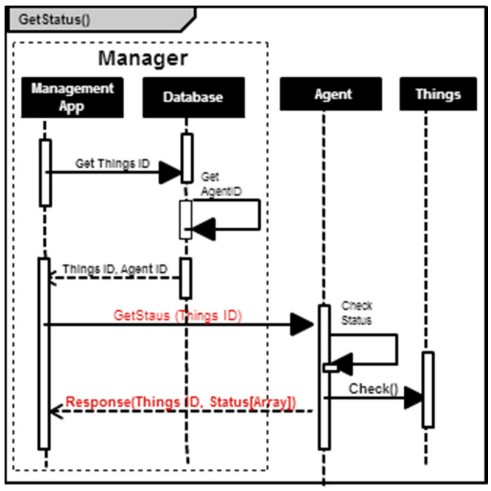


Figure 10: A real-time status request of Node 3 initiated by Node 6 sequence diagram

Table 1- ZigBee configuration Parameters

Channel Type	Channel/Wireless Channel
Radio-propagation model	Propagation/TwoRayGround
Physical Layer Technology	IEEE-802.15.4 PHY
MAC Layer Technology	Mac/802_15_4
Interface queue type	Queue/DropTail/PriQueue
Link layer type	LL
Antenna type	OmniAntenna
Max packet	10
Number of nodes	7
Area of Network Deployment	*50x50-meter square*
Network Layer Routing protocol	AODV
Traffic type used	CBR UDP/FTP

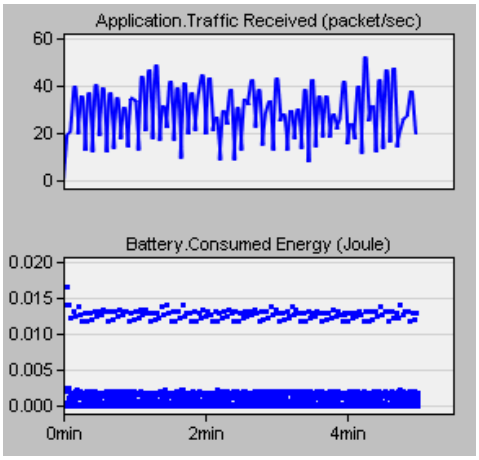


Figure 11: Application Traffic received compared to battery consumption

VI. CONCLUSIONS

This paper presents a fitting platform for management of the IoT, the IoT-MP. The proposed platform provides the capabilities needed for managing things remotely, for instance over the Internet. It is specially designed to work with things, which have limited communication, processing, and power capabilities. It is based on a distributed architecture, utilizing agent and manager paradigm to provide the required functionalities, which are normally part of conventional network management systems. The IoT-MP uses an extensible design where things are defined using attributes in the management database. These attributes are accessible via the IoT-MP and are used by management applications to monitor and control things. In this way, the IoT applications running above the IoT-MP can access things transparently, irrespective of the underlying used communication technologies. It can also eliminate the interoperability issues and facilitate thing-to-thing communications. The proposed platform has also been evaluated through various simulation studies. The results of these studies, reported in this paper, display the advantages of the IoT-MP and its management capabilities for operation within the IoT-based low power wireless networks. Our future works will expand the IoT-MP to include a novel and explicit privacy component, aiming to enhance the location privacy of things within the IoT.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer Networks,* vol. 54, pp. 2787-2805, 2010.

[2] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper,* 2011.

[3] M. Elkhodr, S. Shahrestani, and H. Cheung, "The Internet of Things: Vision & Challenges," in *IEEE Tencon Spring 2013*, Sydney, Australia, 2013.

[4] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security Challenges in the IP-based Internet of Things," *Wireless Personal Communications,* vol. 61, pp. 527-542, 2011.

[5] E. Borgia, "The Internet of Things vision: Key features, applications and open issues," *Computer Communications,* vol. 54, pp. 1-31, 2014.

[6] J. Cooper and A. James, "Challenges for database management in the internet of things," *IETE Technical Review,* vol. 26, pp. 320-329, 2009.

[7] B. Guo, D. Zhang, Z. Wang, Z. Yu, and X. Zhou, "Opportunistic IoT: exploring the harmonious interaction between human and the internet of things," *Journal of Network and Computer Applications,* vol. 36, pp. 1531-1539, 2013.

[8] L. Yang, S. Yang, and L. Plotnick, "How the internet of things technology enhances emergency response operations," *Technological Forecasting and Social Change,* vol. 80, pp. 1854-1867, 2013.

[9] R. H. Weber, "Internet of Things–New security and privacy challenges," *Computer Law & Security Review,* vol. 26, pp. 23-30, 2010.

[10] M. Elkhodr, S. Shahrestani, and H. Cheung, "A Contextual-adaptive Location Disclosure Agent for General Devices in the Internet of Things," in *Local Computer Network (LCN)*, Sydney- Asutralia, 2013.

[11] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks,* vol. 57, pp. 2266-2279, 2013.

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

[13] B. Kalpana and N. S. R. Krishna, "A New Extensible Key Exchange Scheme For Wireless Sensor Networks," *IJSEAT,* vol. 2, pp. 801-805, 2014.

[14] N. E. Petroulakis, I. G. Askoxylakis, and T. Tryfonas, "Life-logging in smart environments: Challenges and security threats," in *2012 IEEE International Conference on Communications (ICC)*, Ottawa, Canada, 2012, pp. 5680-5684.

[15] ZigBee Alliance. (2014). *ZigBee IP and 920 IP*. Available: http://www.zigbee.org/zigbee-for-developers/network-specifications/zigbeeip/