White Paper, First Draft: The design and specification of IoT Mega hub.*

No Author Given

No Institute Given

Abstract. IoT mega hub is a software-based module that acts as the central IoT hub connecting different hubs in the smart home system. The main purpose of introducing the IoT mega hub is to centralise the management of smart home system in case different IoT hubs from different brands are used in the system. IoT mega hub facilities the devices management and control and it also makes smart home system more user friendly. In this paper, we define the IoT mega hub specification including the physical and functional architecture of the system, the communication message protocol between mega hub and other hubs and the security aspect.

Keywords: mega hub; smart home; IoT; security; smart home hub;

1 Introduction

In general, smart home devices are connected to smart home hub, through which user can control and access them. The smart home hubs, existed in the market, are built by different companies (e.g. samsung smartthing, logitech harmony, etc.) [16] based on different access control specifications: different access control models and policy expression languages.

Each designed hub supports a limited number of smart devices as defined by the company that creates the hub. Thus, it is possible that the same smart home system may need to use hubs from different companies if the provided functionalities of one hub is not enough to address the required needs in the home system. Using multiple hubs of different brands in the same smart home system may not be an easy task since each hub provides its own device management console. Using different console to manage devices in a home is not a user friendly system, hence, harmonising the device management console for different hubs can facilitate user in managing those smart devices and make smart home system more attractive. In this paper, we propose the software-based IoT mega hub that acts as the intermediately between client application and smart hubs. This mega hub centralises the management and control of smart devices.

This paper is structured as follows. Section 2 presents the global architecture of smart home system with mega hub integration. Section 3 talks about the functional requirements and global architecture of mega hub. Section 4 is about

* This work is supported by FEDER-IDEE Project.

the communication message format between mega hub and individual smart hub. Section 5 is the conclusion.

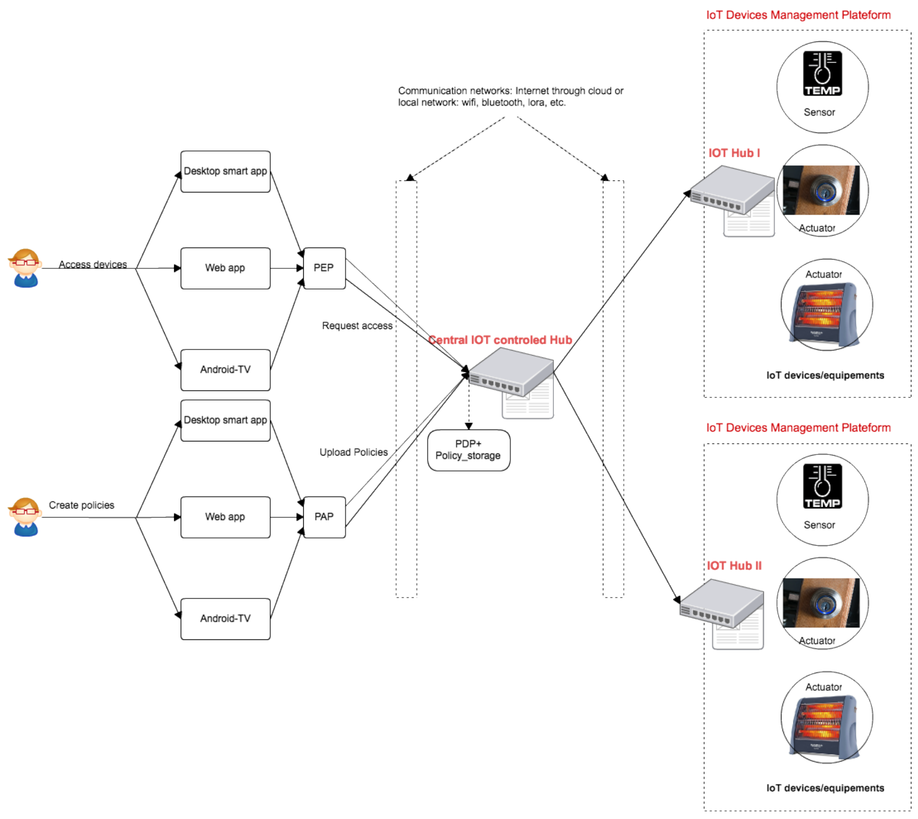

Fig. 1. Smart home architecture with access control management and decision modules

2 Global Architecture: smart home system with mega hub integration

Smart home system is a new technology and it not mature yet [12], most of the existing technologies [16] provides basic smart home functionalities [8] where security of data and devices are basically based on the traditional authentication method (e.g. username and password). For example, smartthing hub [16] developed by Samsung [16] allows users to control devices by mean of user account in Samsung cloud service. The current version of smartthing does not support any customised access control policies definition to reflex user's preference. However, since smart home system is evolving, the future system may provide more options for user to control their devices. Taking into account different aspects of the existing technologies and the future evolution that they might have, we define the global architecture of smart home system where access control management and decision modules are integrated as in Figure 1. The architecture in Figure 1 consists of the following modules.

- IoT device management platform is a network of smart devices controlled by the IoT hub. The devices can be anything such as environmental sensors, actuator, or other type of smart devices.
- IoT smart home Hub, a smart component, allows all devices in the smart home network to securely communicate with each other. This component is also responsible for communicating with IoT service in the cloud through different network technologies for information exchange.
- PDP (Policy Decision Point) is the module responsible for evaluating the access control policies defined by user.
- Policy Storage is a media storage storing access control policies uploaded by PAP. This policy storage is accessible by PDP during policy evaluation phrase.
- PAP is the policy administration module, a console allowing user to define access control policies for devices in the network. Once the policy is created, PAP uploads policy to policy storage.
- PEP (policy enforcement point) is responsible for enforcing the extra obligation that user or system may need to perform before or after the access permission is granted.
- Desktop smart app/ web app/ Android TV are the different types of user's media used to access different smart home modules, such as accessing and managing devices.
- Centralised IoT controlled hub (mega) is a virtual smart home hub used as interface between PEP, PAP and the smart home hubs (hubs produced by different companies). Every access request attempt from client application should go through mega hub where request is processing and forwarded to corresponding smart hub.

We propose this architectures based the assumption that the existing smart home hub does not support access control mechanism and it uses only the traditional authentication method for securing devices and data pertaining to them.

1. Architecture 1 (Figure 1). In smart home context, a fine-grained access control policy is required in order to limit access to different types of user it may have. For example, CCTV can be controlled only by parents and children with certain ages are prohibited to access some TV channels that are not suited for them. Another example, controlling smart door remotely may be prohibited from child given the risk he may create. Given these examples, we can see clearly that there is a need for a fine-gained and different level of access control to smart devices, the traditional authentication is not enough to address this issue. However, most existing smart home systems,

such as smartthing hub [16], logitech harmony [18], google hub [19] do not provide to user the ability to define the complex access control policies for smart devices in home. They use a simple authorisation method by means of user's account to control access to device. To provide that abilities for user to define his own complex access control policies, we need to introduce another module acting as the intermediate entity (central IoT control hub) between the smart home hub and user's application. We propose the smart home architecture with the access control system as presented in Figure 1. In that architecture, every smart home hub[1] in the system is connected to the central IoT control hub. Central hub accesses devices through different hubs by mean of authentication method. The access control decision is performed at central hub. User in the system connects to devices through central hub instead of each hub in the sub network of devices. As shown in Figure 1, user can access device or create access control policies. When accessing smart device, user's access request needs to be sent to PEP module, which forwards the request to PDP module that is generally integrated in the central hub. If the PDP provides positive response, the central hub will connect to sub-network's hub and build a secure channel where user can use later to access devices in that network. In the architecture in Figure 1, user can also define the access control polices, through PAP module. Once the policy is created, PAP uploads the policy to central hub where it is stored.

3 Functional requirements and Global Architecture of Mega hub

In this section, we focus on the functional requirements and mega hub's architecture. The structure of request message is also presented in this section.

3.1 Mega Hub Functional requirements

The main objective of building Mega Hub is to centralise and facilitate user in devices and access control policies management. Furthermore, the mega hub should be able to communicate with other existed smart hubs developed by different companies. Given that, the mega hub should support different communication protocols and security mechanisms to be able to interoperate with different smart hubs. The below functional requirements should exist in the design of our proposed mega hub.

1. Usability. User should be able to perform different settings to allow mega hub to connect to different smart hubs.
2. Security. The communication between mega hub and smart hub should be secure.

[1] This smart home hub is the hub produced by different companies, such as Samsung smartthing, logitech harmony, google hub, etc.

3. Extensibility. Mega hub should be extensible. If new smart hub available in the market, the mega hub can operate with them by just extending its communication interface without going through major modification. This adjustment should be possibly done by ordinary user with a software package update.
4. Access Control. Mega hub should be able to provide the fine-grain access control to devices and data pertaining to them. User should also be able to enable or disable the access control option.

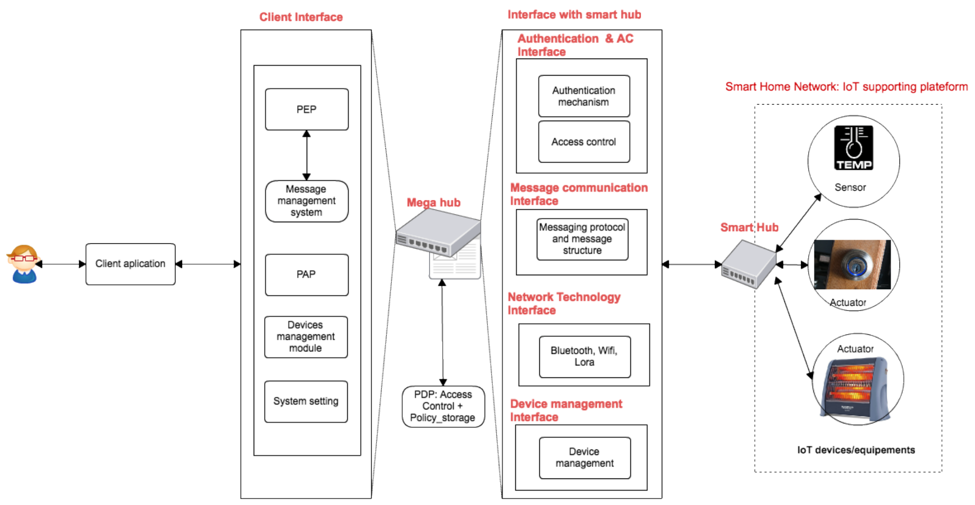
Client Interface
Client aplication
PEP
Message management system
PAP
Devices management module
System setting
Mega hub
PDP: Access Control + Policy_storage
Interface with smart hub
Authentication & AC Interface
Authentication mechanism
Access control
Message communication Interface
Messaging protocol and message structure
Network Technology Interface
Bluetooth, Wifi, Lora
Device management Interface
Device management
Smart Hub
Smart Home Network: IoT supporting plateform
TEMP
Sensor
Actuator
Actuator
IoT devices/equipements

Fig. 2. Mega hub functional architecture

3.2 Mega hub System architecture

As shown in Figure 2, mega hub architecture consists of different components. It can be divided into two main parts. First part of the architecture is for interfacing between mega hub and other smart hubs in the system. The second part is the client interface where user can, through different type of devices (e.g. smart phone, android TV, etc.), connect to mega hub for system setting, device management and access control.

1. Interface with smart hub.
   – Authentication and access control (AC) Interface. This module is responsible for authenticating mega hub with the individual smart hubs. Since different smart hubs may use different authentication mechanisms, this module generally contains different components supporting different authentication methods. If access control mechanism is also required by smart hub, this module should also be able to support.

- Message communication interface. This module is responsible for managing communication messages between mega hub and smart hub. It supports different message management protocols (e.g. MQTT, Websocket, etc.) that are used for exchanging structure data between mega hub and smart hub. This module also contains a sub-module that is responsible for structuring data into a standard message structure (e.g. xml, json, etc.) used for each message management protocol.
- Network technology interface. This module is responsible for managing the network used for the communication between mega hub and individual smart hub. Some wireless technologies such as, blutetooth, wifi and lora are good example of wireless technologies used in IoT system, smart home in particular.
- Device management interface. This module is responsible for managing devices in the network. This module acts as the intermediately between client application and individual smart hub. This is because, the actual device management operation is general performed by smart hub, not mega hub. Mega hub is responsible for only examining the client's request, deciding and finally forwarding the request to individual smart hub. The device management module is generally used to add new devices to the network, remove them, modify device's property and so on.

2. Interface with client application. This module acts as the interface between client application and mega hub. It provides different functionalities such as policy enforcement point, policy administration point, interface for device management and general system setting.
   - Policy Enforcement Point (PEP) is responsible for enforcing the duties that user or system needs to perform before or after access permission to device or data is granted.
   - Message management system is responsible for structuring request/response message in accordance with the standard required by each message management protocols used in mega hub.
   - Policy Administration Point (PAP) is responsible for managing the access control policy. It allows user to create, modify or remove the access control policies defined for devices as well as data pertaining to them.
   - Device management module is a module providing the comprehensive device management interface for user to manage devices in each individual network through mega hub.
   - System setting module is responsible for providing user with comprehensive interface for setting up the mega hub and other modules embedded in mega hub.
3. Client application is a software-based module installed in different smart devices such as, smart phone, tablet, android TV or other devices, which are used to connect to mega hub.
4. Policy Decision Point (PDP) and policy storage. This module has two parts. First of which is the PDP that is responsible for deciding whether to grant or deny access to user's request. The decision is done based on the defined policy in the storage. The second part is the policy storage, which is responsible or

managing the policies created by PAP module. The management includes, how to store, to secure and to retrieve policies.

4 Communication message format between Mega hub and smart hub

In this section, we define the communication messages between mega hub and smart hub. The messages we define in this section are mostly the instruction that mega hub sends to smart hub for execution. There are four different types of instruction: instruction to add device, instruction to remove device, instruction to get data from smart device and instruction to turn-on or off device.

4.1 Add device request (instruction)

In order to add smart device to the network, mega hub needs to send a message containing the information required by each individual hub. The structure of the message should be understood by corresponding smart hub. Below we provide the general message structure where it can be expressed in standard message format such as JSON or XML.

AddDevice
requestId: request identification (data type)
deviceId: device identification (data type)
deviceDescription: description of device (data type)
megaHubId: mega hub identification (data type)
smartHubId: smart hub identification (data type)

Each element in request structure is expressed as follows.

- "AddDevice" is a keyword indicating add device request instruction.
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- deviceId: device identification (data type), deviceId is a key work. device identification is a value of device id. Datatype is a type of device id value (e.g. string).
- deviceDescription: description of device (data type). deviceDescription is a keyword, description of device is a value of device description. Datatype is a type of the device description value (e.g. text).
- megaHubId: mega hub identification (data type). megaHubId is a keyword. mega hub identification is a value of megaHubId. Datatype is a type of the mega hub value (e.g. text).
- smartHubId: smart hub identification (data type). smartHubId is a keyword. smart hub identification is a value of smartHubId. Datatype is a type of the smart hub value (e.g. text).

4.2 Add device response (reply)

Once smart hub receives the “add device” request to add device from mega hub, it executes request and then replies back to mega hub. The response message structure is as follows.

addDeviceResponse
responseId: response id value (data type)
requestId: request identification (data type).
responseCode: responseCode value (data type)

– “addDeviceResponse” is a key word indicating the response message.
– responseId: response id value (data type). “response_id” is a keyword. response id value is a unique value of response id. Datatype is a type of the response id value (e.g. text).
– requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
– responseCode: responseCode value (data type). responseCode is a keyword. Response code value is a unique value of response code. 00000 response code indicating that device cannot be added while 11111 indicates otherwise.

4.3 Remove device request (instruction)

In order to remove smart device to the network, mega hub needs to send a message containing the information required by each individual hub. The structure of the message should be understood by corresponding smart hub. Below we provide the general message structure where it can be expressed in standard message format such as JSON or XML.

RemoveDevice
requestId: request identification (data type)
deviceId: device identification (data type)
deviceDescription: description of device (data type)
megaHubId: mega hub identification (data type)
smartHubId: smart hub identification (data type)

Each element in request structure is expressed as follows.

– “RemoveDevice” is a keyword indicating remove device request instruction.
– requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
– deviceId: device identification (data type), deviceId is a key work. device identification is a value of device id. Datatype is a type of device id value (e.g. string).

- deviceDescription: description of device (data type). deviceDescription is a keyword, description of device is a value of device description. Datatype is a type of the device description value (e.g. text).
- megaHubId: mega hub identification (data type). megaHubId is a keyword. mega hub identification is a value of megaHubId. Datatype is a type of the mega hub value (e.g. text).
- smartHubId: smart hub identification (data type). smartHubId is a keyword. smart hub identification is a value of smartHubId. Datatype is a type of the smart hub value (e.g. text).

4.4 Remove device response (reply)

Once smart hub receives the "remove device" request to remove device from mega hub, it executes request and then replies back to mega hub. The response message structure is as follows.

removeDeviceResponse
responseId: response id value (data type)
requestId: request identification (data type).
responseCode: responseCode value (data type)

- "removeDeviceResponse" is a key word indicating the response message.
- responseId: response id value (data type). "response_id" is a keyword. response id value is a unique value of response id. Datatype is a type of the response id value (e.g. text).
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- responseCode: responseCode value (data type). responseCode is a keyword. Response code value is a unique value of response code. 000000 response code indicating that device cannot be removed while 111111 indicates otherwise.

4.5 Get data from device request (instruction)

In order to get data from smart device in the network, mega hub needs to send a request message containing the information required by each individual hub. The structure of the message should be understood by corresponding smart hub. Below we provide the general message structure where it can be expressed in standard message format such as JSON or XML.

getDataDeviceRequest
requestId: request identification (data type)
deviceId: device identification (data type)
deviceDescription: description of device (data type)
megaHubId: mega hub identification (data type)
smartHubId: smart hub identification (data type)

Each element in request structure is expressed as follows.

- "getDataDeviceRequest" is a key word indicating get data from device request instruction.
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- deviceId: device identification (data type), deviceId is a key work. device identification is a value of device id. Datatype is a type of device id value (e.g. string).
- deviceDescription: description of device (data type). deviceDescription is a keyword, description of device is a value of device description. Datatype is a type of the device description value (e.g. text).
- megaHubId: mega hub identification (data type). megaHubId is a keyword. mega hub identification is a value of megaHubId. Datatype is a type of the mega hub value (e.g. text).
- smartHubId: smart hub identification (data type). smartHubId is a keyword. smart hub identification is a value of smartHubId. Datatype is a type of the smart hub value (e.g. text).

4.6 Get data from device response (reply)

Once smart hub receives the "get data from device" request from mega hub, it executes request and then replies back to mega hub. The response message structure is as follows.

getDataDeviceResponse
responseId: response id value (data type)
requestId: request identification (data type).
responseCode: responseCode value (data type)
deviceStatus: status value (data type)

- "getDataDeviceResponse" is a key word indicating the response message.
- responseId: response id value (data type). "response_id" is a keyword. response id value is a unique value of response id. Datatype is a type of the response id value (e.g. text).
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- responseCode: responseCode value (data type). responseCode is a keyword. Response code value is a unique value of response code. 00000000 response code indicating that device's value cannot be retrieved while 11111111 indicates otherwise.
- deviceStatus: status value. deviceStatus is a keyword. status value is a value of device status indicating the state of each device. In general, there are three types of statuses: On, Off and sleep modes.

4.7 Turn-on device request (instruction)

In order to turn on smart device in the network, mega hub needs to send a request message containing the information required by each individual hub. The structure of the message should be understood by corresponding smart hub. Below we provide the general message structure where it can be expressed in standard message format such as JSON or XML.

turnOnDeviceRequest
requestId: request identification (data type)
deviceId: device identification (data type)
deviceDescription: description of device (data type)
megaHubId: mega hub identification (data type)
smartHubId: smart hub identification (data type)

Each element in request structure is expressed as follows.

- "turnOnDeviceRequest" is a key word indicating get data from device request instruction.
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- deviceId: device identification (data type), deviceId is a key work. device identification is a value of device id. Datatype is a type of device id value (e.g. string).
- deviceDescription: description of device (data type). deviceDescription is a keyword, description of device is a value of device description. Datatype is a type of the device description value (e.g. text).
- megaHubId: mega hub identification (data type). megaHubId is a keyword. mega hub identification is a value of megaHubId. Datatype is a type of the mega hub value (e.g. text).
- smartHubId: smart hub identification (data type). smartHubId is a keyword. smart hub identification is a value of smartHubId. Datatype is a type of the smart hub value (e.g. text).

4.8 Turn-on device response (reply)

Once smart hub receives the "turnOnDevice" request from mega hub, it executes request and then replies back to mega hub. The response message structure is as follows.

turnOnDeviceResponse
responseId: response id value (data type)
requestId: request identification (data type).
responseCode: responseCode value (data type)
deviceCurrentStatus: status value (data type)
devicePreviousStatus: status value (data type)

- "turnOnDeviceResponse" is a key word indicating the response message.
- responseId: response id value (data type). "response_id" is a keyword. response id value is a unique value of response id. Datatype is a type of the response id value (e.g. text).
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- responseCode: responseCode value (data type). responseCode is a keyword. Response code value is a unique value of response code. 00000000 response code indicating that device cannot be turned on while 11111111 indicates otherwise.
- deviceCurrentStatus: status value. deviceCurrentStatus is a keyword. status value is a value of device current status indicating the state of each device. In general, there are three types of statuses: On, Off and sleep modes.
- devicePreviousStatus: status value. devicePreviousStatus is a keyword. status value is a value of device previous status indicating the state of each device. In general, there are three types of statuses: On, Off and sleep modes.

4.9 Turn-off device request (instruction)

In order to turn off smart device in the network, mega hub needs to send a request message containing the information required by each individual hub. The structure of the message should be understood by corresponding smart hub. Below we provide the general message structure where it can be expressed in standard message format such as JSON or XML.

turnOffDeviceRequest
requestId: request identification (data type)
deviceId: device identification (data type)
deviceDescription: description of device (data type)
megaHubId: mega hub identification (data type)
smartHubId: smart hub identification (data type)

Each element in request structure is expressed as follows.

- "turnOffDeviceRequest" is a key word indicating get data from device request instruction.
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- deviceId: device identification (data type), deviceId is a key work. device identification is a value of device id. Datatype is a type of device id value (e.g. string).
- deviceDescription: description of device (data type). deviceDescription is a keyword, description of device is a value of device description. Datatype is a type of the device description value (e.g. text).

- megaHubId: mega hub identification (data type). megaHubId is a keyword. mega hub identification is a value of megaHubId. Datatype is a type of the mega hub value (e.g. text).
- smartHubId: smart hub identification (data type). smartHubId is a keyword. smart hub identification is a value of smartHubId. Datatype is a type of the smart hub value (e.g. text).

4.10 Turn-off device response (reply)

Once smart hub receives the "turnOnDevice" request from mega hub, it executes request and then replies back to mega hub. The response message structure is as follows.

turnOffDeviceResponse
responseId: response id value (data type)
requestId: request identification (data type).
responseCode: responseCode value (data type)
deviceCurrentStatus: status value (data type)
devicePreviousStatus: status value (data type)

- "getDataDeviceResponse" is a key word indicating the response message.
- responseId: response id value (data type). "response_id" is a keyword. response id value is a unique value of response id. Datatype is a type of the response id value (e.g. text).
- requestId: request identification (data type). requestId is a keyword. request identification is a unique value of requestId. Datatype is a type of requestId value (e.g. string).
- responseCode: responseCode value (data type). responseCode is a keyword. Response code value is a unique value of response code. 000000000 response code indicating that device cannot be turned off while 111111111 indicates otherwise.
- deviceCurrentStatus: status value. deviceCurrentStatus is a keyword. status value is a value of device current status indicating the state of each device. In general, there are three types of statuses: On, Off and sleep modes.
- devicePreviousStatus: status value. devicePreviousStatus is a keyword. status value is a value of device previous status indicating the state of each device. In general, there are three types of statuses: On, Off and sleep modes.

5 Conclusion

In this paper, we present the first draft of the design of mega hub. The presentation includes: the global architecture of the smart home system with mega hub integration, the functional requirements and global architecture of mega hub and the communication message format between mega hub and individual smart hub. Our next step is to design the detailed mega hub architecture for the implementation. The choice of technologies including network technologies and messaging protocols are also included in our future work study. We also envisage to develop a complete software-based mega hub that can be used to connect different network technologies, specifically, lora, wifi and bluetooth.

References

1. Bruce Ndibanje, Hoon-Jae Lee and Sang-Gon Lee. Security Analysis and Improvement of Authentication and Access Control in the Internet of Things. Open access Sensors, 14(8), 14786-14805; doi:10.3390/s140814786, 2014.
2. Rahul Godha, Sneh Prateek and Nikhita Kataria. Home Au- tomation: Access Control for IoT Devices. International Journal of Scientific and Research Publication, Volume 4, Issue 10, October 2014, ISSN 2250-3153.
3. Blase Ur, Jaeyeon Jung and Stuart Schechter. The current State of Access Control for Smart Devices in Homes. Workshop on Home Usable Privacy and Security (HUPS), July 24-26, 2013, Newcastle, UK.
4. Ricardo Neisse, Gary Steri, Igor Nai Fovino and Gianmarco Baldini. SecKit: A Model-based Security Toolkit for the Internet of Things. The journal of Computer and Security (2015), page 60-76. Published by ELSEVIER.
5. Sachin Babar, Parikshit Mahalle, Antonietta Stango, Neeli Prasad and Ramjee Prasad. Proposed Security Model and Threat Taxonomy for the Internet of Things. International Conference on Network Security and Applications (CNSA 2010). Recent Trends in Network Security and Applications pp 420-429. Published in Springer 2010.
6. J. Sathish Kular and Dhiren R. Patel. A survey on Internet of Things: Security and Privacy Issues. International Journal of Computer Applications. Volume 90. No 11, March 2014.
7. Assessement of Access Control Systems. National Institute of Standards and Technology. Technology Administration U.S. Department of Commerce. http://csrc.nist.gov/publications/nistir/7316/NISTIR-7316.pdf
8. Smart Home service providers. http://www.sensorsmag.com/components/top-10-smart- home-service-providers-us
9. Definition of cloud service. http://www.webopedia.com/TERM/C/cloud services.html
10. Agrawal, Rakesh and Imielin?ski, Tomasz and Swami, Arun. Mining Association Rules Between Sets of Items in Large Databases. SIGMOD Rec. June 1, 1993. Vol 22, No 2., New York, NY, USA.
11. Extensible Markup Language (XACML). http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html
12. M. Schiefer. Smart Home Definition and Security Threats. Ninth International Conference on IT Security Incident Man- agement IT Forensics. pages. 114-118, May 2015.
13. Jose L. Hernandez-Ramos, Antonio J. Jara, Leandro Marin and Antonio F. Skarmeta. Distributed Capability-based Access Control for the Internet of Things. Journal of Internet Services and Information Security (JISIS), volume: 3, number: 3/4, pp. 1-16.
14. N. Komninos and E. Philippou and A. Pitsillides. Survey in Smart Grid and Smart Home Security: Issues, Challenges and Countermeasures. IEEE Communications Surveys Tutorials. vol. 16, No. 4, P. 1933-1954, 2014.

15. Wireless Technologies. https://www.link-labs.com/blog/types- of-wireless-technology
16. Samsung smart thing . https://www.smartthings.com/
17. One M2M specification. http://www.onem2m.org/
18. Logitech harmony. http://www.logitech.com/en- us/product/harmony-hub
19. Google hub. https://on.google.com/hub/