Industry-Friendly Engineering Tools for Wireless Home Automation Devices

Jia Wang[1], Zhibo Pang[2], Cheng Pang[3], and Valeriy Vyatkin[3, 4]
[1]Royal Institute of Technology (KTH), Stockholm, Sweden [2]Corporate Research, ABB AB Vasteras, Sweden.
3Aalto University, Finland; [4]Luleå University of Technology, Sweden
jiaw@kth.se; pang.zhibo@se.abb.com; {cheng.pang.phd, vyatkin}@ieee.org

Abstract—**Although home automation (HA) systems in the wired domain are widely accepted by consumers, in today's industry, the mega trend is steering HA systems along a wireless way. Theoretically, wireless solutions are able to provide HA systems with more flexibility and thus reducing engineering costs. In practice, however, deploying wireless HA systems actually requires more costs and efforts due to the lack of versatile software tools to support the whole engineering process. This paper defines and evaluates the engineering workflow and architecture for home automation systems. The proposed architecture is studied and implemented based on web technologies and graphical configuration environments, with the aim of reducing workloads of HA engineers at every stage. A prototype has been implemented to demonstrate the technical feasibility of the proposed architecture.**

I. INTRODUCTION

An automation system for residential buildings or homes is one of the most promising application areas of the Internet-of-Things (IoT) [1]. State-of-the-art residential houses are distinguished not only by the sustainability of architecture, but more importantly by the automation functionality [2, 3]. Thanks to the development and flexibility of wireless technologies, wireless solutions have brought business opportunities for device manufacturers due to the vast demands for products in the wireless domain [4]. As a result, wireless solutions, such as EnOcean [5], ZigBee [6], and 6LoWPAN [3], are gaining momentum these days. However, this rapid development is hampered by the unavailability of appropriate engineering tools [7]. Thus, the wireless HA industry is demanding a set of engineering tools to standardize the engineering process [8] and address issues, such as extensive manual workloads [9] and terrible interoperability among different wireless technologies [10].

Significant amount of efforts have been devoted to the development of engineering tools and workflow for HA systems. To simplify the installation of HA devices, some standards are going to integrate commissioning tools into their products, such as EnOcean and ZigBee. But, there is a lack of specification for common engineering tools and device description files that can be shared among the value chain. A more customer-oriented commercial solution is the Apple HomeKit framework [11]. It proposes a MiFi license where HA devices supporting this license can be seamlessly integrated with iOS devices [11]. However, the mobility of iOS devices makes HomeKit fragile. When the device is out of home, the original smart home may immediately lose control. Similarly, as an industry-oriented solution, DeviceCloud is a typical example for a cloud-based management platform for IoT connected devices, which combines service provisioning, devices real time control, and devices data visualization to simplify the process of bringing the interaction of products of any type [12]. However, its defect in console units fails DeviceCloud to become an industry-friendly HA system. For discovering and pairing new devices, authors of [13] proposed an XML-based secure integration approach to avoid loading additional drivers. To enhance the system interoperability, authors of [14] presented a technology integrated platform that is capable of interfacing with building managing systems. Other efforts are made on the application layer based on the Model-View-Controller concept [15-17] and on the network layer by deploying heterogeneous sensor nodes with uniform interfaces [18]. These interfaces are controlled by a Web based central control unit [19]. Furthermore, to enhance users' entertainment experiences, heterogeneous services are integrated with a goal-driven approach. Once a user declared his requirements, the system will automatically search for and set up available services to maximally satisfy the user's goal [20]. In short, versatile engineering tools are required for supporting the whole engineering process.

In this paper, a wireless HA system architecture is developed to efficiently integrate and engineer (e.g. pre-configuration, configuration, and commission) heterogeneous wireless devices. As an industry-friendly engineering tool, it facilitates works for all roles in the HA industrial value-chain, including end users and installers who have limited programming or system skills, system integrators with little configuration skills of wireless networks, and device manufacturers providing products with various protocols. Moreover, since all these roles in the engineering process are indispensable to implement the final system, the business benefits of the entire HA value chain has to be well protected by the proposed architecture.

The remainder of this paper is organized as follows. In Section II, we elaborate the engineering workflow of HA systems. In Section III, we explain the architecture and technical details of the proposed tools. Finally, the implementation details and experiment results are given in Section IV. The paper is finally concluded in Section V.

978-1-4799-6648-6/15/$31.00 ©2015 IEEE

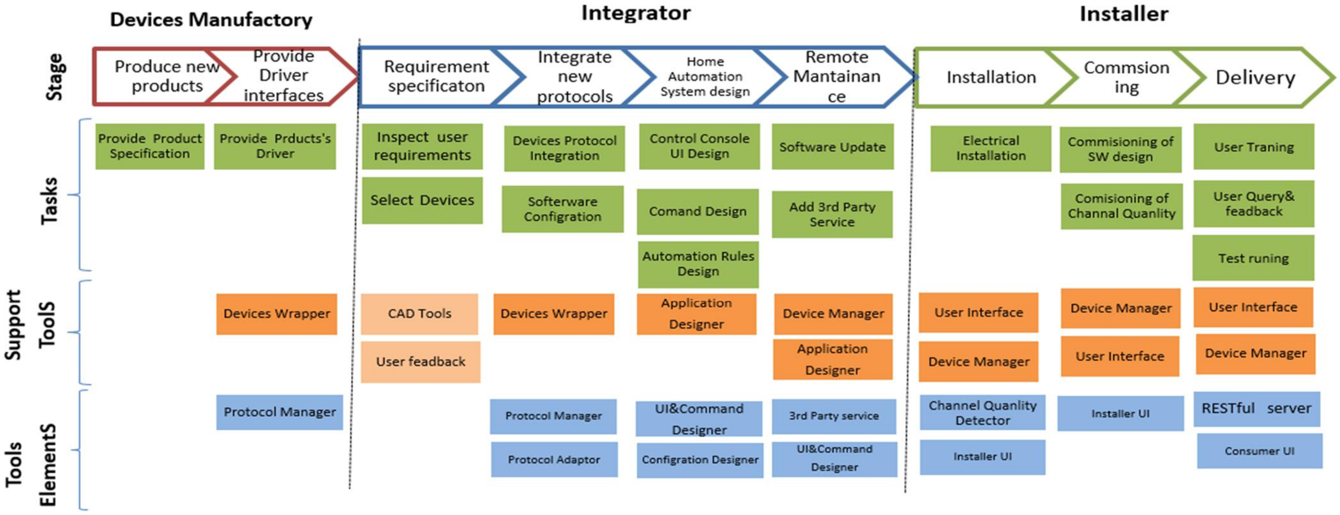


Fig. 1. Engineering Workflow of Home Automation Systems.

II. Engineering Workflow of Home Automation Systems

In practical engineering, implementation of wireless HA systems is accomplished by an engineering workflow [10] where device manufacturers, integrators, and installers are all involved. Fig. 1 illustrates the tasks at each stage and the tools that are needed.

At first stage, device manufacturers usually provide integrators the product packages, including specifications and device drivers. Depending on the project scope, integrators usually divides his work into four parts. The first part is to generate a project specification, which specifies the functionalities to realize and the HA devices to select [4]. As long as drivers are included in their product packages, in the second stage, integrators will integrate these devices into the wireless HA system. Therefore, an integration tool will be very helpful, if loading drivers to the device gateway and modifying configuration system, automatically. Then, the integrator is about to generate a user interface (UI) panel for controlling the HA system. Thus, the tool is needed, in this stage, enable to customize UI design as well as to configure the system on site. System remote maintenance will be a task for integrators in the last stage, if system updating is needed in the future.

When the integration work is done, installers will deploy the HA system to the user's home. During installation, the wireless signal preparation is a complicated issue. Therefore, the framework providing a visual installation environment will facilitate this process, by visualizing data such as working channel, signal strength, and transmission rate on the UI panel. After the installation, the HA system has to be fully tested before being delivered to end user.

III. Proposed Home Automation System Architecture

A. *The overall HA system architecture*

The overall HA system architecture is outlined in Fig. 2. The key elements of the Smart Home system included in the WiHA system are separated into three domains as follows. More details are available in our previous work [2, 3].

- Automation Device Domain: It comprises all the field devices including the sensors and actuators, wireless sensor and actuator network (WSAN) devices (e.g. routers, bridges), home automation gateway, and Consumer User Interface (UI) (e.g. smart phone with apps).
- IT Service Domain: It comprises the backbone system of various IT services (e.g. energy analytics, home service, utility billing, and entertainment), public cloud infrastructure, in-home IT gateway, and Consumer UI.
- Automation Service Domain: It provides internal services for configuring and updating HA system.

B. *The deployment of proposed tools in the HA system*

The proposed configuration tools are deployed in the above domains. In particular, the Device Wrapper is a set of embedded software executed in the Device Gateway, the Device Manager is a set of embedded software executed in the System Access Point, the Application Designer is an application software executed on the server of system integrator, and User Interface (UI) is a set of application software executed on the user devices both for installers and end consumers. The functionality of each tool is introduced below and more details are given in the next sub-sections.

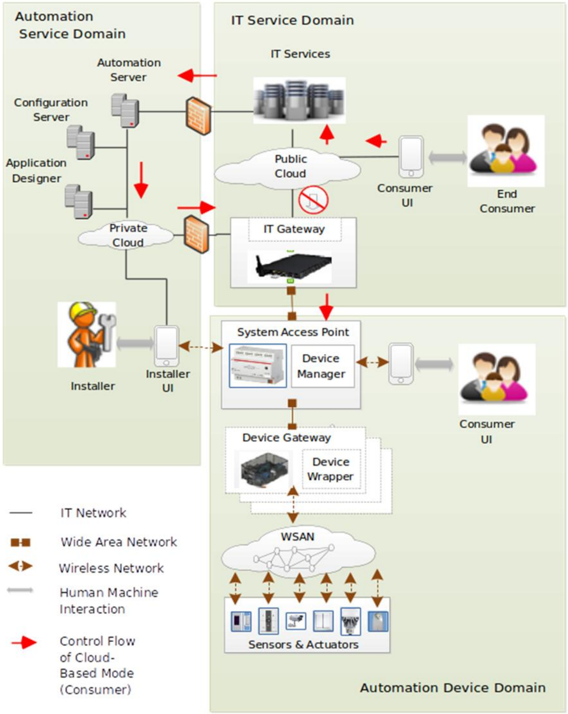


Fig. 2. Home Automation System Overview.

- Device Wrapper. The Device Wrapper is loaded with different HA devices' driver in handling the physical connection with the HA devices.

- Device Manager. The "always on" Linux/Windows PC runs an instance of the Device Manager. As the brain of HA system, the Device Manager controls the HA system by forwarding commands from the consumer UI panel to the specified URL. Moreover, the execution sequence of the commands can be ruled and scheduled by its sequencer engine.

- Application Designer. The Application Designer is a web application, utilized to configure the control function for the system. This tool contains a graphical UI, with which integrators can easily create a control panel for the HA system online. Once the design for a specific control panel is made, this result will be saved as an XML document in the database. Moreover, the Application Designer also supports to maintain the HA system and update software remotely, thus, facilitating integrators by reducing their travel time between office and costumers.

- User Interface. When connected to the Devices Manager, the user-end controller UI application is updated automatically and immediately, reflecting changes made by the Application Designer.

C. Device Wrapper

As illustrated in Fig. 3, Device Gateway implements the server, running the Device Wrapper. Device Wrapper is the bridge between the Devices Manager and a set of devices integrated in the home WSAN. Device Wrappers encapsulate and declare all devices' drivers APIs as public APIs. Hence, the Device Manager is able to issue commands as well as to read data from HA devices by sending REST/HTTP requests forwarded by the Device Gateway. This tool consists of four elements as follows:

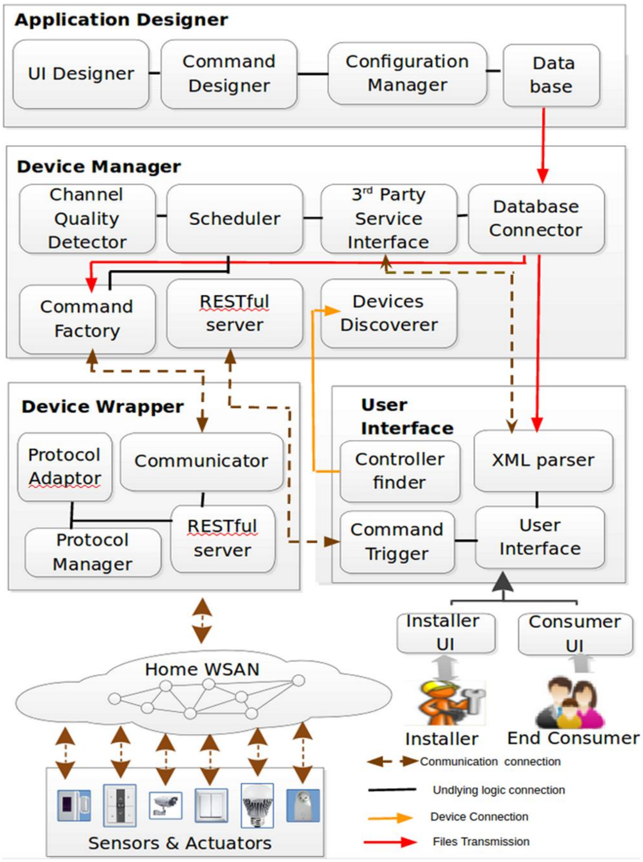


Fig. 3 Home Automation System Architecture.

- Protocol Manager: As various HA devices need to be integrated in the HA system, Protocol Manager enables integrator to easily manage and configure drivers loaded in the Devices gateway. Protocol Manager provides a unique ID for each driver API. The API parameters are configured from the XML element under a particular ID tag.

- Protocol Adaptor: The Protocol Adaptor takes the role as a translator for the Device Wrapper by parsing XML information. In order to expose driver interfaces as resources for the RESTful server, the Protocol Adaptor maps driver interfaces with the corresponding URL registered in the XML document. As a result, the HA device driver instances can be easily searched and built by the RESTful server.

- RESTful server: The RESTful server handles all REST/HTTP requests from the Device Manager and then it builds a specific interface out of a bunch of interfaces of drivers. Hence, when interacting with a particular device, at the high level, the application may simply send a command using "GET" or "POST" method. At the low level, the function related device drivers will be built and communicate to the particular serial port to enable hardware functions.

D. Device Manager

As mentioned in the previous section, the Device Manager is the central control unit of the HA system. The following list shows technical details in this tool:

- The Device Discoverer sets up a connection between the Device Manager and the customer UI application. It supports device auto-discovery by listening for application requests on certain ports. Moreover, the multicast group IP address is applied to secure such connection. When sending connection requests to a specific port, only the device with the authenticated IP address can set up the TCP connection with a particular Device Manager.
- Command Factory: The output of Application Designer provides each function command a unique ID, which is mapped with the particular URL of the RESTful server implemented in the Device Gateway. This file is a XML-based document saved in the database. After being synchronized with the database, the Command Factory is able to extract information and combine commands with URLs listed in XML elements. Finally, the command factory turns these combinations into Java object instances as REST resources for the RESTful server.
- The Database Connector is responsible for setting up the connection between the Device Manager and the database in the Application Designer. After being confirmed and synchronized with the database, all resources such as XML documents and UI images will be downloaded to the Device Manager.
- Channel Quality Detector: Some existing network tools (e.g. Kismet) can be used to monitor wireless channel quantity and the diagnosing results are saved in XML-based documents. To realize the diagnosis function, Channel Quality Detector is responsible for extracting signal strength and channel information from the XML document and making them available for the RESTful server.
- Scheduler: This module is responsible for arranging commands on the basis of the pre-set orders and executing them at a particular time, e.g. wake up HA owner early when it rains.
- 3rd Party Service Interface: Enables the Device Manager to get access to data and integrates services provided by 3rd party companies, such as the geographical position information and weather forecast information
- Communicator: Handles communication between Device Manager and other components. By using the “GET” or “POST” method, this module is responsible for sending REST/HTTP requests to a specific module.

E. Application Designer

Usually, configuring and commissioning are two tough tasks for non-technician users to complete. The Application Designer is to bridge the gap between design flexibility and the underlying technology requirements. Moreover, design results can be saved as reusable templates, which significantly improve the engineering efficiency.

- Command Designer: This module is a handy tool to design each UI element (e.g. buttons and labels) with a particular function triggering command. This module provides each command with a unique ID mapping with a particular HTTP request URL of the Device Wrapper. Finally, function designs and related mapping relations are represented in the Devices Specification File as an XML-based document saved in the database.
- Configuration Manager: This module provides wizard steps with which the integrator can follow to pre-set configuration options for HA devices. The pre-set configuration options then will reflect on the installer UI panel. For example, in a complex HA project, paring switches with each actuator is usually a headache work for installers. Thus, the installer UI panel, that enables installers to change HA devices paring relationship on site, will improve the engineering efficiency.

F. User Interface

The Customer UI is a remote control application working on mobile devices. It allows consumers to remotely control HA devices. The technical details are listed below:

- Controller Finder: As mentioned above, the Device Manager listens on the multicast group to discover client devices. Therefore, the Controller Finder is responsible for adding IP addresses of potential client devices into the multicast group in the Devices Manager.
- Customer UI: When setting up the Customer UI, this application may firstly send a REST/HTTP request to the Device Manager, from which to load the UI Specification File. After parsing this XML-based document to modify the HTML file, the Customer UI is reflected on the user console panel. Moreover, the Customer UI is also able to display data received from sensors via the Communicator module.
- Command Trigger is responsible for listening user’s actions (e.g. press a button) and then triggers commands (i.e. turn on the light) by sending a request to a specific URL based on the Device Specification File. In addition, this module also acts as a reception for data from sensors or 3rd party services.

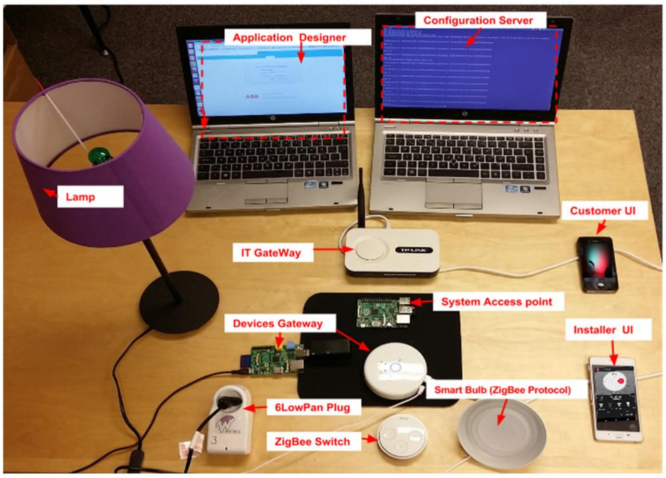


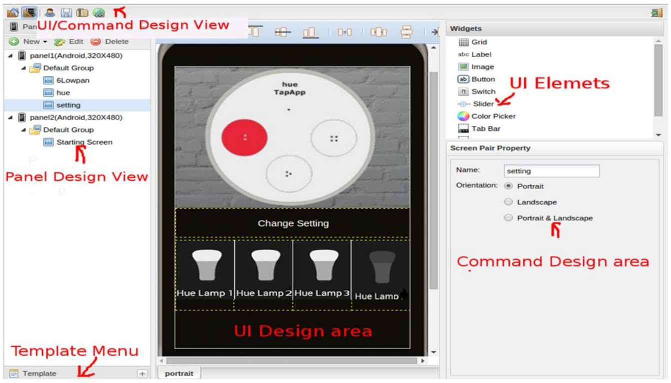
Fig. 4. The Prototyping System.

IV. Implementation And Demonstration

A. *Implementation Details*

In order to evaluate the proposed system, two performance tests have been experimented. In this the first experiment, 6LoWPAN plugs from Watteco NKE Electronics and ZigBee bulbs from Philips Hue are selected to demonstrate the integration capability of the proposed system. In this case, the HA gateway needs to be equipped with a 6LoWPAN USB Broad Router that enables the HA gateway to access the 6LoWPAN plugs. Part of the hardware setup is shown in Fig. 4.

Raspberry Pi, a low cost embedded Linux PC which has 512 MB of RAM and a 100Mbps Ethernet port, takes the role of the HA gateway using the Raspbian Image as the operation system. The Device Manager acts as the HA gateway add-in, running the OpenRemote open source code [21] with the size of 144.8 MB. As an add-in for the device gateway, the Device Wrapper is written by the Python language, and its RESTful server is implemented by the Bottle RESTful server framework with the total size of 48 MB.The database is based on SQLite database running on the Raspberry Pi. The Designer is implemented by the OpenRemote Web application (73.5 MB) mounted on the Apache server in another PC. Finally, the command scheduling is implemented based on the open Door event processing language.



Name: setting
Orientation: ● Portrait ○ Landscape ○ Portrait & Landscape

Fig. 5. Screenshot of the Designer Application User Interface.

When logging into the application, at the start-up, the following options are available for users to choose: UI designer view or command designer view. The former one enables user to craft the console panel by simply dragging and dropping design elements onto the “panel” area (Fig. 5). The latter one assists users to bind UI elements with command functions that have been configured in the “window” view.

Fig. 6 shows the snapshot of how the user operates console panel to get into the “configuration mode”. Firstly, the user needs to select a particular IP address of a Device Manager hardware out of the given list. Then, the user may choose the panel identity, whereby the application designed before will be reflected on the mobile device (Fig. 6 right). In the configuration mode, the controlled object of the physical switch can be changed by clicking the identity button of the Hue light.

B. *Performance Tests*

In order to evaluate the proposed system, two performance tests have been experimented. In this the first experiment, the system performance is valued from Trigger Command Round Trip Time (RTT). Trigger Command RTT is a latency result of a simple closed loop, which starts from the moment that the system access point issues a command to Device Gateway and ends up when the system access point gets response from the server of Device Gateway. Similarly, in the second experiment, Execute command RTT is measured. It starts from the moment that Devices Gateway sends command to a particular HA device and ends up when receiving response from the HA device.

The data captured from the experiment are plotted in Fig. 7. In this experiment, we used the 6LoWPAN plug as an example to measure the latency of the wireless HA system. In this system, all devices shown in Fig. 4 are set up and connected to a local office network.

From the plotted data (Fig. 7), it is obviously shown that Execute command RTT is much longer than Trigger command RTT. This suggests the system latency is mainly caused by the data transmission delay, i.e. the time it takes to transmit packages in the 6LoWPAN network, rather than the data processing delay and the command execution delay caused by the system.

In the second experiment, “turning on” commands are continuously issued to the 6LoWPAN plug. From the statistics of the execute command RTT listed in Table 1, it is shown

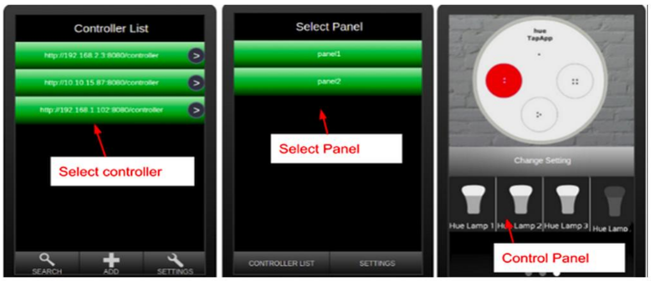


Fig. 6. Screenshot of the Controller Application User Interface.

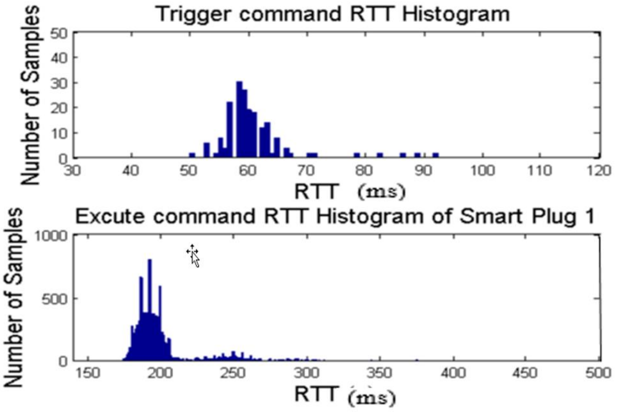


Fig. 7: Screenshot of the Console Application User Interface.

that the average time the 6LoWPAN plug responses to the driver with the 6LoWPAN network ranges from 174.0ms to 650ms. Meanwhile, the average time that the Device Wrapper acquires HTTP commands from the Device Manager and triggers the corresponding driver ranges from 50.0ms to 35.3ms. The above results show that, when controlling different HA devices, the main limitation of the system response time is caused by the latency of HA devices' sub-network, i.e. 6LoWPAN network where the wireless plug has connected to.

V. Conclusion

Efficient HA engineering tools is a crucial precondition for the further growth of wireless HA industry. This paper defines and evaluates the engineering workflow and engineering for home automation. Then, the architecture of an industrial friendly HA system is proposed, with the aim of reducing workloads of HA engineers at every stage. Finally, a proof-of-principle minimal system is implemented based on web technologies. Preliminary result of functional test confirms the feasibility of the proposed tools and indicates response time of system is acceptable for home automation. And this latency could be further reduced if optimizing the response performance of the HA devices' sub-network.

Table 1. Statistics of Round Trip Time (RTT) of the HA System.

	Execute Command RTT	Trigger command RTT
Min RTT (ms)	174.0	50.0
Average RTT (ms)	206.9	64.3
Max RTT (ms)	650.0	198.0
Sigma(ms)	39.5	35.3

References

[1] Z. Pang, "Technologies and Architectures of the Internet-of-Things(IoT) for Health and Well-being," Doctoral thesis, School of Information and Communication Technology, Royal Institute of Technology (KTH), Stockholm, Sweden, 2013.

[2] Z. Pang, Y. Cheng, M. E. Johansson, and G. Bag, "Preliminary Study on Industry-Friendly and Native-IP Wireless Communication for Building Automation " in *1st International Conference on Industrial Networks and Intelligent Systems (INISCom 2015)*, Tokyo, Japan, 2015, p. accepted.

[3] Z. Pang, Y. Cheng, M. E. Johansson, and G. Bag, "Preliminary Study on Wireless Home Automation Systems with Both Cloud-Based Mode and Stand-Alone Mode," in *17th IEEE International Conference on Computational Science and Engineering (CSE 2014)*, Chengdu, China, 2014, pp. 970-975.

[4] J. Ploennigs, H. Dibowski, U. Ryssel, and K. Kabitzsch, "Holistic Design of Wireless Building Automation Systems," in *16th IEEE International Conference on Emerging Technologies & Factory Automation (ETFA 2011)*, Toulouse, France, 2011, pp. 1-9.

[5] Veris Industries."EnOcean Alliance defines intelligent commissioning of smart buildings". News Release July 2014.

[6] K. Bong Wan and J. Seong-Soon, "A New Commissioning and Deployment Method for Wireless Sensor Networks," in *3rd International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2009)*, Sliema, Malta, 2009, pp. 232-237.

[7] H. Dibowski, J. Ploennigs, and K. Kabitzsch, "Automated Design of Building Automation Systems," *IEEE Transactions on Industrial Electronics,* vol. 57(11), pp. 3606-3613, 2010.

[8] S. Runde, A. Heidemann, A. Fay, and P. Schmidt, "Engineering of Building Automation Systems — State-of-the-Art, Deficits, Approaches," in *15th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2010)*, Bilbao, Spain, 2010, pp. 1-8.

[9] S. Runde and A. Fay, "Software Support for Building Automation Requirements Engineering—An Application of Semantic Web Technologies in Automation," *IEEE Transactions on Industrial Informatics,* vol. 7(4), pp. 723-730, 2011.

[10] S. Wang, J. Xing, and P. Wang, "Flexible Integration of BAS of buildings in Service and under construction " in *2nd 2011 International Conference on Electrical and Control Engineering (ICECE)*, Yichang, China, 2011, pp. 4514-4518.

[11] Apple Inc. (2015). Homekit [Online]. Available: https://developer.apple.com/homekit/

[12] Etherios. (2015). DeviceCloud [Online]. Available: www.etherios.com/products/devicecloud

[13] D. Kümper and R. Tönjes, "Remote Configuration and Deployment of Sensor Drivers for a Medical Bluetooth Sensor Gateway " in *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Lucca, Italy, 2011, pp. 1-6.

[14] M. S. Aslam, A. Guinard, A. McGibney, S. Rea, and D. Pesch, "Wi-Design, Wi-Manage, Why Bother?," in *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, Dublin, Ireland, 2011, pp. 730-744.

[15] L. Barta, D. Boyle, B. O'Flynn, and E. Popovici, "Simplified Commissioning and Maintenance for Wireless Sensor Networks: a Novel Software Tool," in *26th International Conference on Architecture of Computing Systems (ARCS 2013)*, Prague, Czech Republic, 2013, pp. 1-6.

[16] C.-H. Yang, V. Vyatkin, and C. Pang, "Model-Driven Development of Control Software for Distributed Automation: A Survey and an Approach," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 44(3), pp. 292-305, 2014.

[17] C. Pang, V. Vyatkin, Y. Deng, and M. Sorouri, "Virtual Smart Metering in Automation and Simulation of Energy-Efficient Lighting System," in *18th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2013)*, Cagliari, Italy, 2013, pp. 1-8.

[18] D. Adolf, E. Ferranti, and S. Koch, "SmartScript - A Domain-Specific Language for Appliance Control in Smart Grids," in *3rd IEEE International Conference on Smart Grid Communications (SmartGridComm 2012)*, Tainan, Taiwan, 2012, pp. 465-470.

[19] J. Cecilio and P. Furtado, "Architecture for Uniform (Re)Configuration and Processing Over Embedded Sensor and Actuator Networks," *IEEE Transactions on Industrial Informatics,* vol. 10(1), pp. 53-60, 2014.

[20] S. Mayer, N. Inhelder, R. Verborgh, and R. Van de Wallet, "User-friendly Configuration of Smart Environments," in *2014 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM)*, Budapest, Hungary, 2014, pp. 163-165.

[21] OpenRemote [Online]. Available: http://www.openremote.org