sustainability

MDPI

Article

Consistent Registration and Discovery Scheme for Devices and Web Service Providers Based on RAML Using Embedded RD in OCF IoT Network

Wenquan Jin and Dohyeun Kim *

Department of Computer Engineering, Jeju National University, Jeju 63243, Korea; wenquan.jin@jejunu.ac.kr
* Correspondence: kimdh@jejunu.ac.kr

Received: 28 September 2018; Accepted: 27 November 2018; Published: 10 December 2018

Abstract: The Internet of Things (IoT) is comprised of connected devices which are equipped with sensors, actuators, and applications to provide services for enabling those objects to connect and exchange data. For these connected devices and the IoT network environment, heterogeneous protocols and frameworks have been published and applied to provide novel IoT services in our daily life. On the Internet, most of the services are provided by existing web service providers which are based on the high-performance processor, storage, and stable power supply. However, devices of the IoT are developed for constrained environments using the small size of equipment to provide seamless services ubiquitously. For accessing the constrained devices and existing web service providers using the clients in the IoT networks such as smart homes, the services of servers from the devices and web service providers shall be discovered by the clients using a consistent discovery service. In this paper, a consistent registration and discovery scheme is proposed for the devices and web service providers in the Open Connectivity Foundation (OCF)-based IoT network. For supporting the proposed scheme, an embedded resource directory (RD) server is proposed to provide a consistent registration service that is used for publishing information of devices and web service providers. For the registration, a unified profile format is used that is based on the RESTful API Modeling Language to describes the information of devices and web service providers. Furthermore, the discovery service provides the consistent interface to discover the registered devices and web service providers by the client using the unified user interface. Accordingly, the client can access the resources of devices and web service providers based on the discovered information.

Keywords: Internet of Things (IoT); resource directory (RD); RESTful API modeling language (RAML); Open Connectivity Foundation (OCF); web service provider (WSP); constrained network

1. Introduction

The Internet of Things (IoT) is an emerging engineering paradigm to build heterogeneous industrial systems for segments such as healthcare, residents, transportation, manufacture, and agriculture. IoT services are provided to the professionals, clients, and providers through the Internet-connected devices at the edge of networks. In the center of networks, powerful computers support sufficient storage and computing ability to enable data saving and processing for smart and ubiquitous applications. In the IoT, the devices are a most important element, which is comprised of information and communication technologies (ICT) to implement the IoT systems in a specific domain or cross-domain of industries. With the development of ICT, the quantity of Internet-connected devices will be increased to reach 20.4 billion by 2020, and the quality also will be enhanced with the deployment of the next generation (5G) communications networks [1,2]. Therefore, the heterogeneity of devices shall be the challenge because of the increasing number of devices [3]. Moreover, most of the

existing services are developed for the hypertext transfer protocol (HTTP)-based web service provider (WSP) on the Internet [4]. However, the IoT frameworks prefer the IoT-specific protocols such as constrained application protocol (CoAP), message queuing telemetry transport (MQTT), and other communication implementations based on those protocols [5]. The IoT system shall support the HTTP-based existing services of WSPs as well as support the constrained and novel IoT-specific protocols. For accessing those massive number of devices in various protocols, the directory service is needed to provide a consistent service to discover the information of objects in the network.

In the IoT network, the directory service enables things to be discovered. The functionality of the directory service supports the information of IoT devices to be stored and retrieved [6]. The resource directory (RD) is a server that provides a set of services for enabling the services of discovery in the network where the RD is deployed [7]. The registration and discovery services are the main features of the RD, which are used for registering the information of services to the directory and providing the discovery service to enable the discovery of accessible services by the clients. Services are exposed by the resources to handle the requests from clients and communicate the results to clients [8]. For registering those services' information to the RD, a profile format is required to include the information. The RESTful application programming interface (API) modeling language (RAML) is used for describing the APIs from a service provider such as an IoT device and WSP. The RAML is a framework to describe the APIs using a structured and unambiguous format [9]. RAML data includes the basic information of the service provider, and the introduction of each resource that includes resource uniform resource identifier (URI), request requirements such as query parameters, and request body structure, response body structure, and examples. Therefore, the RAML data can be used for describing the information of resources and registering the resource to the RD.

In the IoT network, the server applications are deployed on the devices which have a processor, memory, storage, and network communication module. A server application provides the services which are exposed by the resources in the network [10]. The resource discovery approaches can be used with a multitude of techniques according to the number of parameters such as network scale, deployed devices, and communication protocols. In the IoT network, the devices can be deployed in the constrained networks using the constrained network protocols with limited power supply and computing ability [11]. Moreover, some devices need a stable power supply and sufficient storage to provide services using the professional parts such as e-Health sensors in the wireless body area network (WBAN) [12–15]. These devices can be the standalone ubiquitous devices to support real-time decision making through the analysis base on bit data, and also can be the local servers which provide discovery, registration, and forwarding service to the devices and service clients in a specific network [16,17]. However, in the IoT network, the constrained devices and local servers need to collaborate for providing better services to the clients.

In this paper, we present a registration and discovery scheme for the IoT devices and WSPs which are deployed in different networks using different protocols. For the IoT devices, the IoTivity framework is applied, which is an open source implementation of the Open Communication Foundation (OCF) specification [18]. In the OCF network, devices use OCF messages to communicate, which are based on CoAP for the constrained IoT network. The functionalities supported by the RD server provide service of registration, discovery and request forwarding from the OCF network and WSPs from the Internet. The registration service is used by the client that registers the information of discoverable resources. The client can be a publishing client as well as a part of IoT device. An IoT device can hold the resources to provide sensing and actuating services in the OCF network. A WSP can be developed by an organization to provide services through the APIs to enable users to use the APIs and access services. To use the APIs in the OCF network by IoT clients, the information must be discoverable. The proposed RD server provides a consistent registration service for the RAML-based resource information. For the registration, a new data model is presented using the RAML definition. The RAML definition shall be pre-defined for the IoT device as well as the WSP and deployed on the client that can publish the RAML data to the RD server. A parser in the RD server interprets the RAML

data and inserts them into the database (DB). In the OCF network, the IoT clients request the RD server to look up the registered information, then the resources shall be discovered from the OCF network and Internet. Once the resources are discovered, the IoT client can access the services. The RD server also provides the message forwarding service to the OCF client for accessing the HTTP-based services on the Internet. Figure 1 shows the network architecture that presents the interactions of entities for the registration, discovery, and accessing.

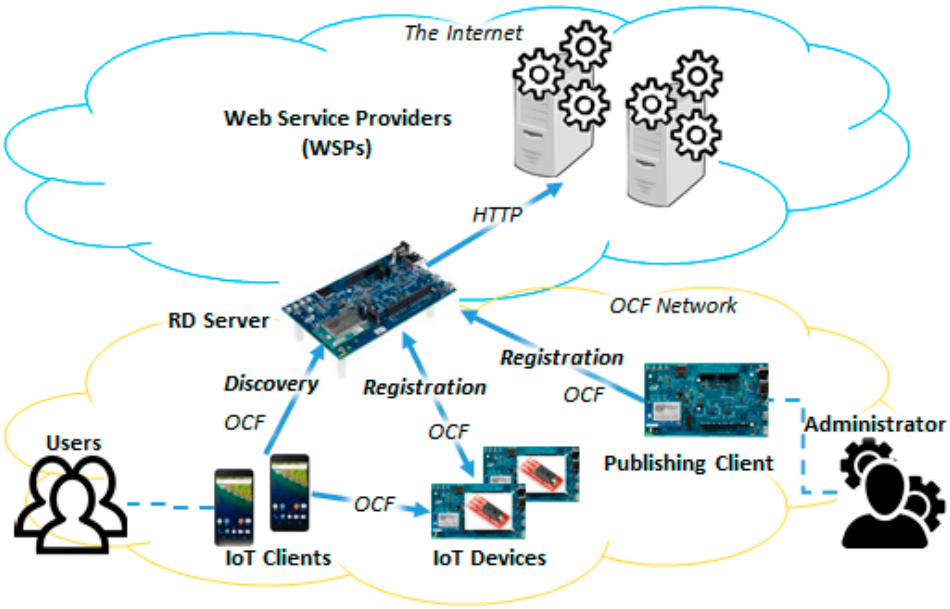

Figure 1. The proposed system architecture.

The rest of the paper is structured as follows. Section 2 introduces related works. Section 3 introduces the proposed methodology of registration and discovery. Section 4 introduces the registration and discovery scenarios using the proposed elements in the OCF-based IoT network. Section 5 introduces the implementation results. Section 6 introduces the performance evaluation. Finally, we conclude this paper in Section 7.

2. Related Works

For the constrained network, solutions of discovery can be considered as distributed and centralized for the IoT environment [19]. The discovery servers can be deployed for the large-scale distributed system that has multiple systems to handle the network where the server is deployed [20]. This fully distributed architecture can support an efficient discovery approach in low power and lossy networks such as the CoAP network [21,22]. The CoAP network is a constrained network because the CoAP is designed for the M2M services which are provided by the constrained devices [23]. The constrained RESTful environments (CoRE) RD is a server that is deployed in the IoT network for providing services to discover the constrained nodes [24]. For the registration to the CoAP RD, the RD provides an interface to accept a POST from an CoAP device containing the list of resources in the CoAP message payload. The resources are described in the CoRE link format to be added to the RD. Therefore, the limitation of the CoAP RD is the registration interface of a CoAP RD and cannot involve the extra information for describing the details of registering devices. For constrained devices, the registration information shall be smaller. However, according to the edge computing, the devices quip with high-performance parts in the IoT network. Therefore, the CoAP RD shall provide the registration interface for the detail registration profile that can describe more detailed information of the registering device.

The server shall be based on the connected power supply to provide the lookup services for retrieving the registered device information that can involve URIs, accessing requirement, and status [25]. Therefore, the RD also can be used for the centralized solution with powerful equipment for large-scale IoT networks [26]. With the sufficient hardware specification, the RD also can support high-performance computing to provide intelligent services based on a large amount of data. The implementation of RD in the high-performance computer can be supported by the personal computer, cloud services, and virtualization solutions [27]. However, many small embedded boards are published for the high-performance such as Raspberry Pi, Intel Edison, and other devices which have a small size with the full ability to support sufficient computation and network communication [28]. Therefore, the embedded boards also enable deployment of the RD server for providing registration and discovery services in the constrained environment.

The web-based service discovery solutions are built with a high-performance computer to provide searching services, such as web searching engines on the Internet. However, traditional web-based searching solutions cannot provide the service in the same way for the IoT network because of the heterogeneity and constrained network requirements [29]. In the IoT network, the discovery service provider shall support resource discovery regardless of the communication protocols and technologies used by IoT clients [30]. However, it is a challenge to fulfill all the communication interfaces for the heterogeneous devices. There are many IoT standard solutions to support the discovery mechanism in the framework level [31–33]. However, most of them present the discovery architecture for their specific communication protocol and service accessing mechanism [34]. The proposed solution is based on the OCF specification that refers to the CoAP-based service providing architecture. The proposal of RD in the OCF specification is focused to provide registration and discovery services for the OCF devices in the OCF network through CoAP communications.

Also, the proposed RD enables the IoT devices and WSPs together to be discovered by the IoT clients in the OCF network. To provide the discovery service using the consistent scheme in order to discover the resource from the HTTP and OCF network, we present a new data model using RAML. In the OCF specification, the data model is presented for the registration of OCF devices [35]. However, the properties of the data model are not sufficient for other IoT framework and WSPs. For example, the presented data model of OCF cannot describe the required parameters of a URI using the properties. The service-oriented architecture (SOA) enables the system development through the separated APIs which are used for accessing the services with the exposed URIs. Many IoT frameworks are based on the SOA to provide the services from the devices [36]. Therefore, the RAML-based description of resources from the IoT network and Internet can be the information in the RD for the discovering by the clients.

3. Proposed Methodology of Registration and Discovery

The proposed network architecture is comprised of IoT devices, WSPs, RD server, publishing client and IoT clients. For the registration process, the IoT devices and publishing client send the RAML data to the RD server. Then, the information of IoT devices and WSPs can be discovered by IoT clients through the RD server. Using the discovered information, the IoT clients can access the services which are provided by the servers in the constrained network and the Internet. Figure 2 shows the proposed architecture of a consistent registration and discovery scheme. Each component represents the role of the entity in the proposed system.

In the RD server, the RAML definitions are saved in the storage of the server. The data of RAML is published by devices and the publishing client. The information of saved RAML definitions can be retrieved by IoT clients. The RD server is comprised of resources and storage which provide a registration service and discovery service to the IoT client and service providers in the OCF network and Internet. The functions of the component write and read the service provider information from the service information storage. The registration service is a function that is used by devices and

publishing client to publish service provider information. The discovery service is a function that is used by clients for obtaining the service provider information.

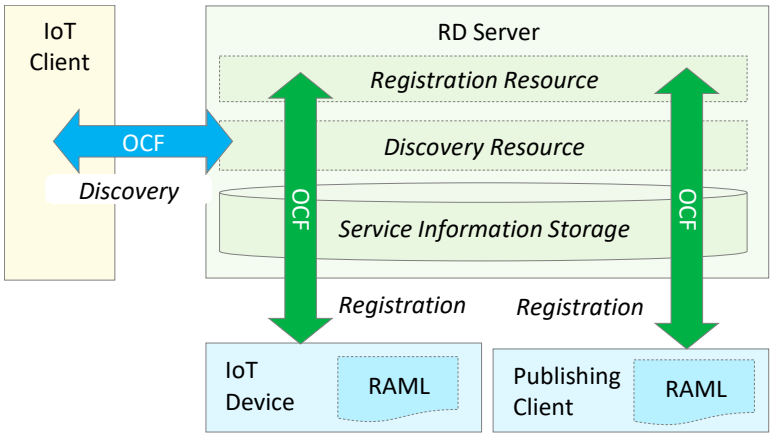

Figure 2. Proposed architecture of consistent registration and discovery scheme.

The system includes the IoT client and publishing client for accessing the RD server. The IoT client is used for requesting to the RD server for discovering the registered information, and the publishing client is used for publishing the information of the HTTP service provider to register the provider's information that is deployed on the Internet. The discovery is a function that is a part of the client. Users can, through the discovery service, find the service provider information by requesting the RD server. The publishing client supports the publishing function. The HTTP service provider can be the application that is run by other organizations or companies. Therefore, we do not have permission to add some functions to their application such as a publishing function. The publishing client reads the profile of the service provider and sends it to the RD server for registration.

In the service providers, the applications can be used for providing services to clients, e.g., sensing services, actuating services, and web services from the Internet. The OCF service provider is used for providing sensing and actuating services that are hosted on IoT devices to support IoT services. The HTTP service provider is used for providing web services such as weather services, social network system (SNS) services, stock market services, etc.

For providing services to multiple clients, the RD server includes the handlers, which handle the requests from the client. In the proposed OCF network, the RD server supports the storage and retrieval functions for the information of IoT devices and WSPs. The server includes the OCF server functionality to provide OCF services using OCF resources. Figure 3 shows the functional architecture of the RD server. The server has the RD resource and RD list resource. The resource name of RD resource is/rd which has a GET handler and POST handler, and the resource name of the RD list resource is/rd/list which has a GET handler. Through these handlers, the OCF server handlers OCF requests. The handlers interact with the DB in the RD server which involves the information of the provider, resource, method, and parameter. In the RD resource, the GET handler is used for handling the discovering request from the IoT client. The response message includes the detailed information of a provider. The POST handler is used for handling the registration request from the publishing client and IoT device. The registration request includes the information of IoT device and WSP. Once the server received the registration request, the server parses the request message and stores to the DB. In the RD list resource, the GET handler is used for handling the discovering request from the IoT client. The response message includes the provider information list.

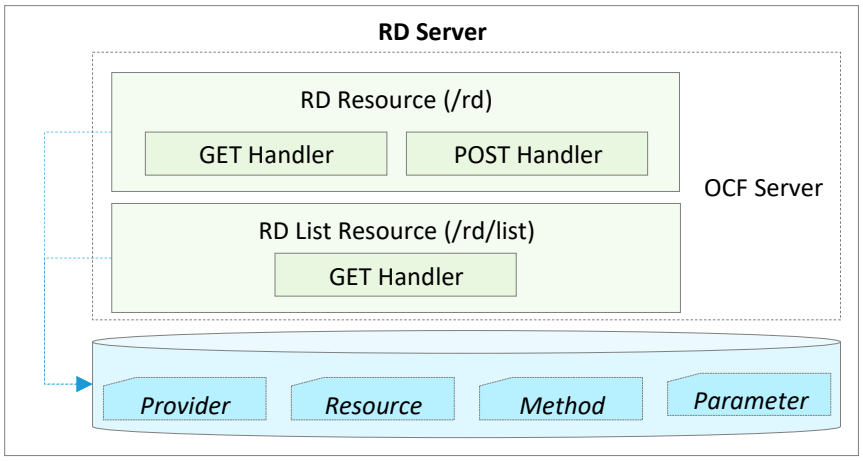

Figure 3. Functional architecture of resource directory (RD) server.

Figure 4 shows the functional architecture of the IoT device. The IoT device is a device that supports functions for sensing from the environment, and through actuators to update the environment where the devices are deployed. The devices may need to be constrained and wireless using the battery and wireless communication equipment such as WiFi, bluetooth low energy (BLE) and long-term evolution (LTE). The IoT device can be developed using IoT boards such as Raspberry Pi, Intel Edison board and Arduino Uno. These IoT boards support processor, storage and communication parts. For building an IoT device, the sensors and actuators are also required. The IoT device includes the OCF server to provide OCF services for sensing or actuating. In the IoT device, the OCF server includes resources for providing OCF based services to clients. The handler of a resource can be implemented for interacting with the sensors or actuators to collect the sensing data or update the environmental parameters such as temperature, humidity and illumination. The device also includes the OCF client to request RD server for registering the information. For registering the device to the RD server, the device sends the information to the RD server through the OCF network. The registration message includes the RAML definition which is used for describing the device. The RAML file can be deployed in each device. Once the device is started, then the device can register the information to the RD server by sending the RAML data.

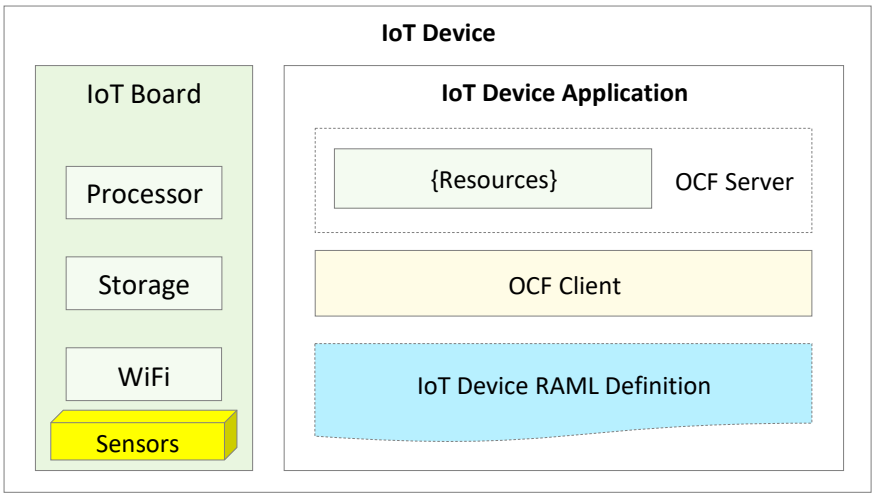

Figure 4. Functional architecture of Internet of Things (IoT) device.

Figure 5 shows the publishing client functional architecture. The architecture includes the software part and hardware part. The software part shows the application and platform for developing the publishing client. The hardware part shows the device information regarding the publishing client.

For developing the publishing client, we use the Android Things platform on the Intel Edison board. The board includes processor, storage, and WiFi modules. The publishing client application includes the OCF client and HTTP service provider RAML definition. The OCF client is used for sending the registration message to the RD server. The HTTP service provider RAML definition is a file that includes the information of the service provider.

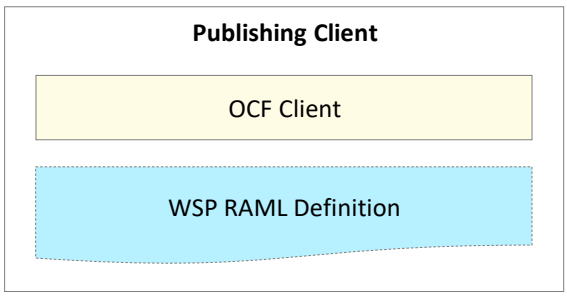

Figure 5. Functional architecture of publishing client.

Figure 6 shows the IoT client functional architecture. The architecture includes the software part and hardware part. The software part shows the application and platform for developing the IoT client. The hardware part shows the device information regarding the IoT client. For developing the IoT client, we use the Android platform on the Android smartphone. The Android phone includes processor, storage, and WiFi module. The IoT client application includes the views and OCF client. The views are used for displaying the information to the user. The provider information list page is used for displaying the provider information list. The provider information detail page is used for displaying the detail provider information of the selected item. The OCF client is used for sending the discovering message to the RD server.

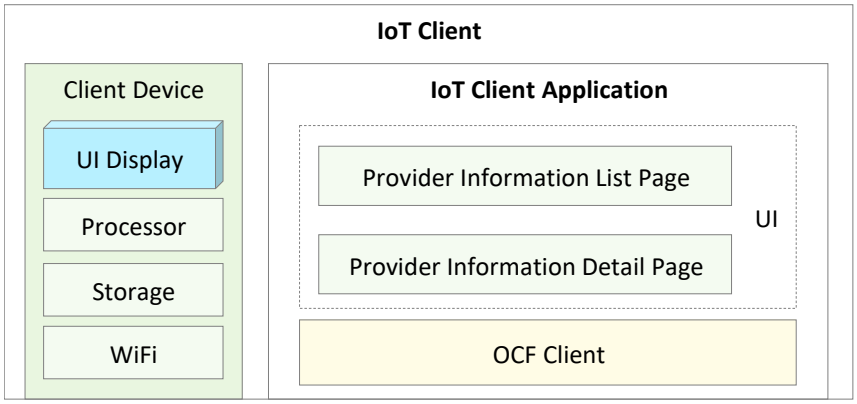

Figure 6. Functional architecture of IoT client.

4. Registration and Discovery Scenario

In this section, the following sequence diagrams illustrate the scenarios of registration and discovery using the elements in the proposed IoT network which are the RD server, IoT device, publishing client, IoT client, and user. The registration process involves the interactions between IoT device and RD server, and between publishing client and RD server. The discovery process is comprised of IoT client, RD server and user.

Figure 7 shows the sequence diagram for registering the IoT device to the RD server. The IoT device is an OCF entity that runs the OCF server for providing the services based on its resources. For starting the server, the device read and write the data to initialize the application and configuration settings. The device information is written in the RAML file that is used to send the device information to the RD server through the OCF communication network. Firstly, the IoT device reads the RAML file

to get the RAML data. Then the IoT device gets name and data from the RAML data. The name and data are included in the payload of the request for the registration. The IoT device sends the message that uses the POST method with URI coap://{RD's IP:5683}/rd and the payload. Once the RD server receives the message, then the server parses the RAML data and inserts them in the DB.

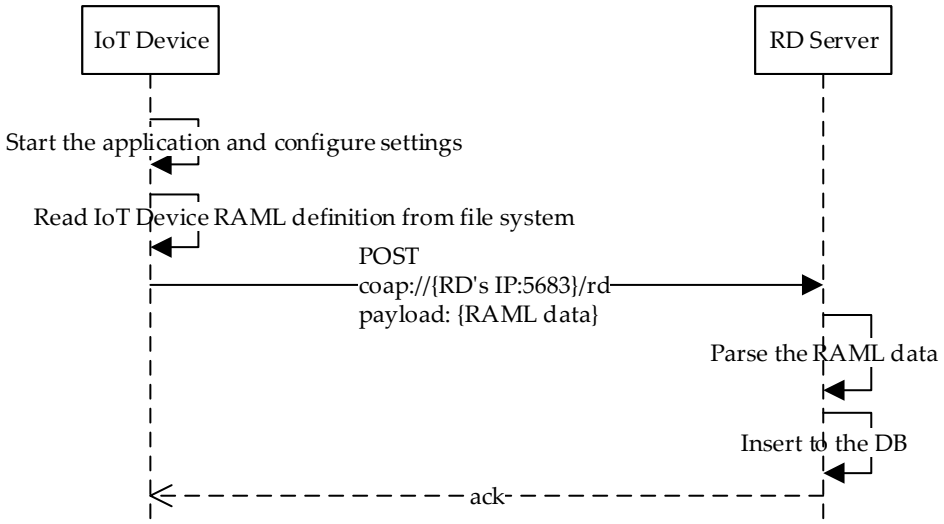

Figure 7. Sequence diagram for self-registration of IoT device.

Figure 8 shows the sequence diagram for registering WSPs which are deployed in the Internet to provide the service based on the HTTP. The information is registered by the publishing client through the OCF communication network. The publishing client is also an OCF device that is used send the RAML data to the RD server through the OCF network. Because the RD provides a consistent interface for the registration, the RAML need to be sent by the same publishing scheme with the IoT device. Therefore, the process is same as the process of registering an IoT device. Firstly, the publishing client reads the RAML file to get the RAML data. Then, the publishing client gets the name and data from the RAML data to put in the payload of the OCF request message. The publishing client sends the message that uses the POST method with URI coap://{RD's IP:5683}/rd and the payload. Once the RD server receives the message, the server parses the RAML data and inserts it to the DB.

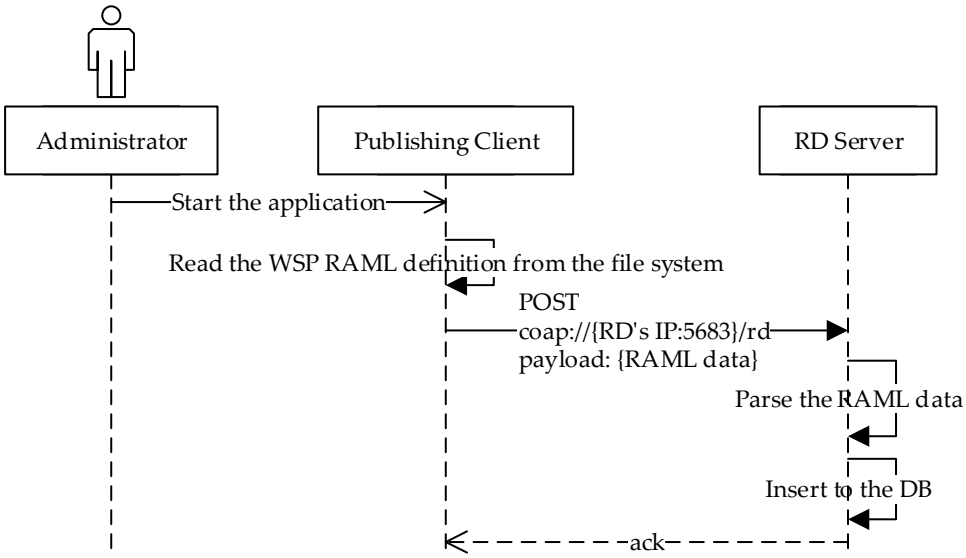

Figure 8. Sequence diagram for registering Hypertext Transfer Protocol (HTTP)-based web service provider (WSP).

Figure 9 shows the sequence diagram for discovering the IoT device and WSP through the RD server. The returned payload includes the information of OCF service providers as well as the HTTP service providers from the OCF network and the Internet. The discovery service is used by the IoT client that requests to the/rd/list resource of the RD server with the keyword query parameter using the GET method. Once the resource of RD server is requested, the handler responds registered provider information list. Each item involves the table of provider's information in the response payload. Once the response message is delivered to the IoT client, the client displays the provider information list. The user needs to click an item from the list. Once the item is clicked, the IoT client requests to the/rd resource of the bridge device with id query parameter using GET method. The handler of the resource responds with the provider information in detail. The data includes all information that relates to the provider.

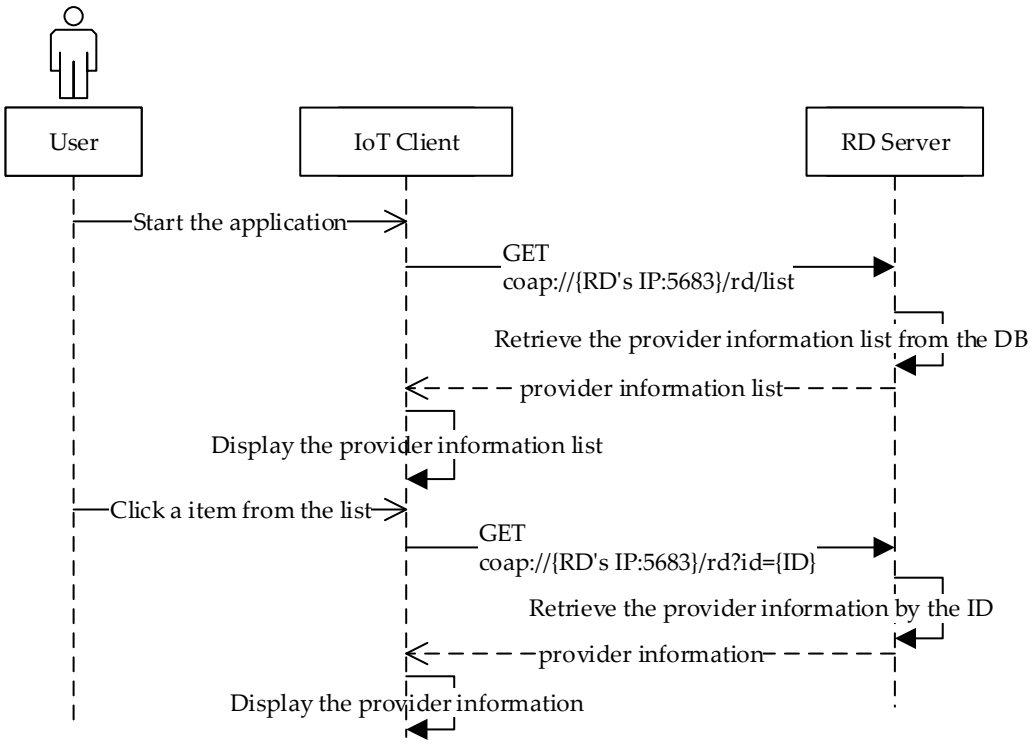

Figure 9. Sequence diagram for discovering IoT device and WSP through RD server.

5. Implementation Results

Table 1 introduces the development environment of the proposed system. The RD server, IoT device, and publishing client using the Intel Edison board for the device, and the IoT client uses the Samsung Galaxy S4 Android smartphone as the device. For the Intel Edison board, the runtime operating system (OS) is Android Things 0.2 that is built by compiling software development kit (SDK) 25 and min SDK 24. For the smartphone, the runtime OS is Android 5.0 Lollipop that is built by compile SDK 25 and min SDK 21. The development tool Android Studio 2.3.1 is used for the Android applications. For the Intel Edison board application implementation, we use IoTivity library that is compiled on Ubuntu 16.4 64 bit for Android OS on x86 CPU. For the smartphone application implementation, we use IoTivity is compiled on Ubuntu 16.4 64 bit for Android OS on armeabi central processing unit (CPU). For the RAML parser, we use RAML parser library that supports RAML version 0.8 and 1.0.

Table 1. Development environment.

Component	RD server	IoT Device	Publishing Client	IoT Client
Physical device	Intel Edison Board			Samsung Galaxy S4
Runtime OS	Android Things 0.2 (Build: compile SDK 25, min SDK 24)			Android 5.0 Lollipop (Build: compile SDK 25, min SDK 21)
tool	Android Studio 2.3.1			
Library and Frameworks	IoTivity 1.2.1(x86), raml-parser-2 1.0.13, jackson-core 2.9.0			IoTivity 1.2.1(armeabi), jackson-core 2.9.0

Figure 10 shows the IoT device's RAML definition. The RAML definition illustrates the service information of the IoT device. According to the RAML definition, the IoT device provides 2 services through the resource/led and resource/temperature. The/led resource has GET and PUT handler for handling the request. The/temperature resource has GET handler for handling the request. For each handler of resources in this IoT device, the query parameters and response body are defined. The response body is defined using JavaScript object notation (JSON) format and an example also is included in this RAML definition.

Figure 10. IoT device RESTful API modeling language (RAML) definition.

Figure 11 shows the HTTP service provider RAML definition. The RAML definition illustrates the service information of HTTP service provider in the Internet. According to the RAML definition, the HTTP service provider provides one service through the resource/weather. The/weather resource has a GET handler for handling the request. For the handler of a resource in this HTTP service provider, the query parameters and response body are defined. The response body is defined using JSON format and an example also is included in this RAML definition. The HTTP service provider is a

weather service provider that provides several weather-related services using open APIs. The service is described in the presented RAML definition that provides the current weather information. Once the service is accessed, the current weather information is returned to the client. The information is also described in the RAML definition.

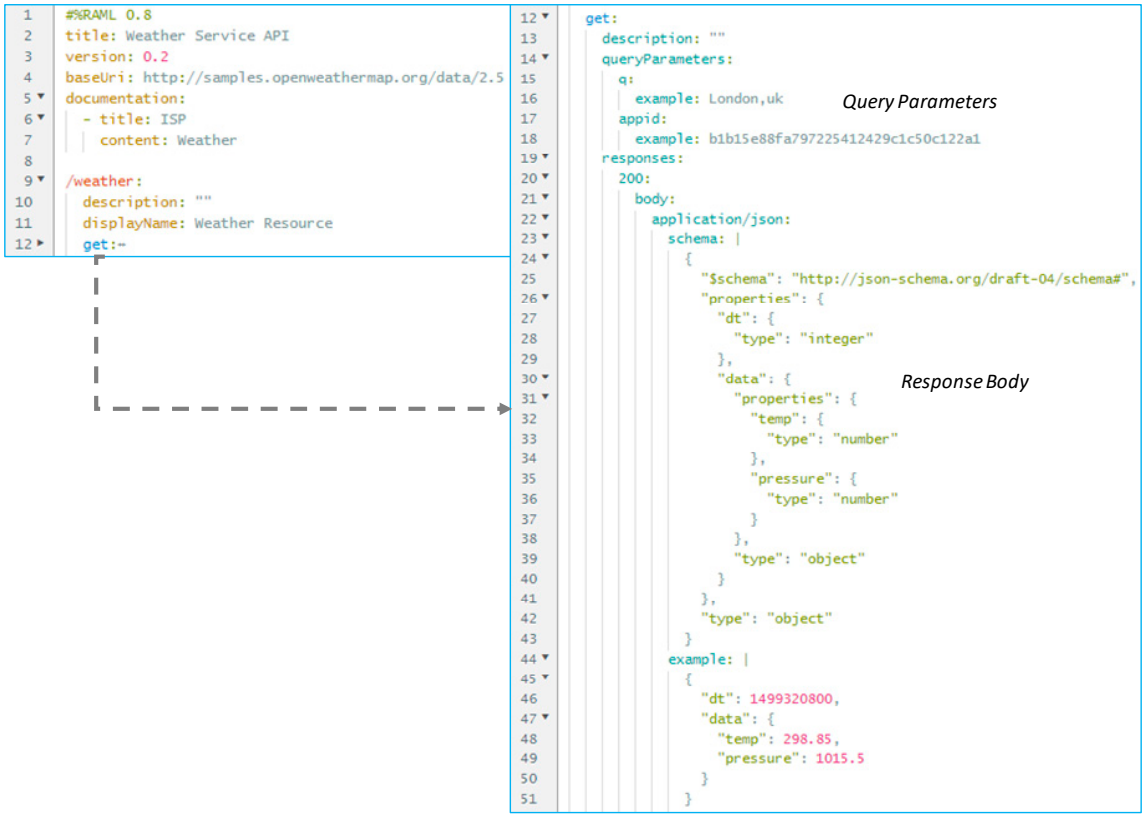

Figure 11. WSP RAML definition.

Figure 12 shows the result of IoT device registration that is captured from the loggings in the RD server. The IoT device sends the information to the RD server using OCF based on CoAP. The request uses the POST method with the payload that includes the RAML data for the information of the IoT device. The RD server is an OCF server that has the RD resource for handling the request for the registration. The OCF is implemented using the IoTivity framework; therefore, some loggings are shown in the results by the IoTivity internal functionalities. The title and information of the RAML definition are printed out in the figure. The title is the IoT service and the print outed string is the name of the RAML file that consists of the title. The RAML data is defined for the IoT device, which is parsed by the RD server to a JSON data, and is used for inserting the DB. The print outed data is the JSON data that includes information from the RAML. The IoT device has resource/humidity that requires parameters rt and if.

Figure 13 shows the result of publishing client registration that is captured from the loggings in the RD server. The publishing clients send the information to the RD server using OCF based on CoAP. The request uses the POST method with the payload that includes the RAML data for the information of the HTTP service provider. The title is Weather Service and the print outed string is the name of the RAML file that consists of the title. The RAML data is defined for the HTTP service provider, which is parsed by the RD server to JSON data, and is used for inserting the DB. The print outed data is the JSON data that includes information from the RAML. The HTTP service provider has resource/weather that requires parameter APPID and q.

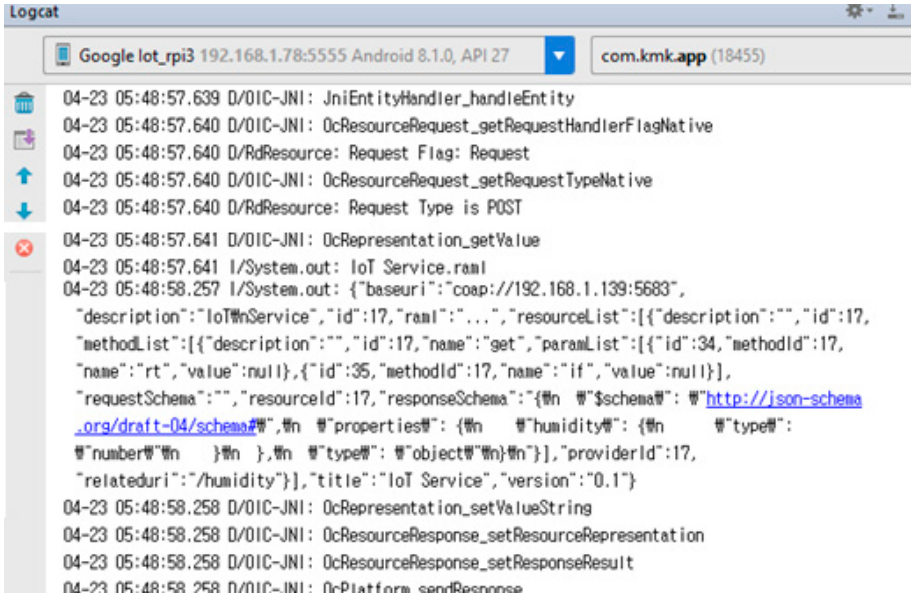

Figure 12. IoT device registration result in RD server.

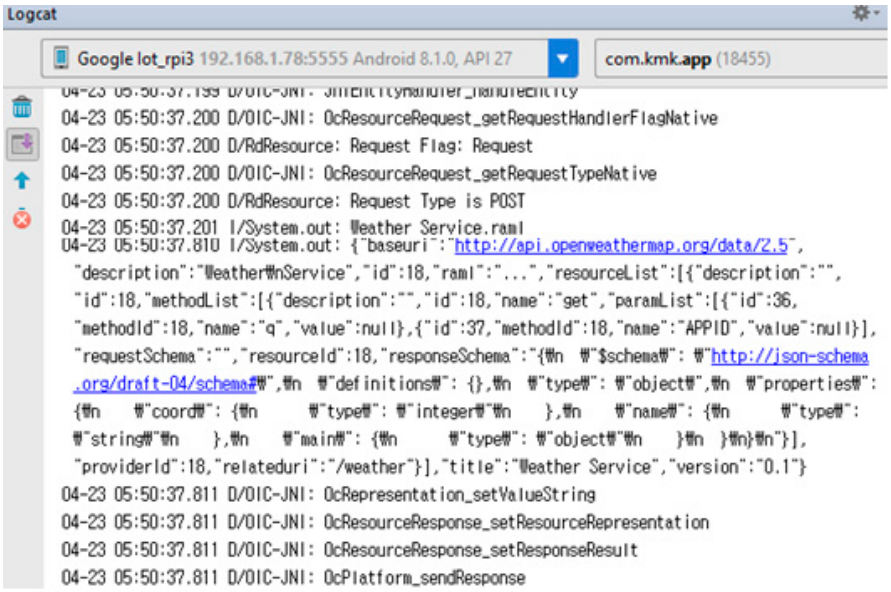

Figure 13. Publishing client registration result in RD server.

Figure 14 shows the result of displaying the service providers' information in the list page and detail page of the IoT client. The page displays the registered service providers information list. Each item of the list displays the RAML name, version, and description. For requesting the page, the IoT client needs to include query parameters in the request URI. The parameter is the OCF resource interface, and the parameter keyword is used to retrieve the list with the query. The parameter startRowNo is used for pagination, and parameter pageSize is used for pagination. The request method is GET, and the handler of the method is in the RD server that is used for getting the service provider information list. The detail page displays the registered service provider detail information. The page includes the resource list because a provider can include multiple resources. For requesting the page, the IoT client needs to include query parameters in the request URI. The parameter is on the OCF resource interface, and id is used for retrieving a provider information by its ID. The request method is GET, and this method handler is in the RD server that is used for obtaining service provider information.

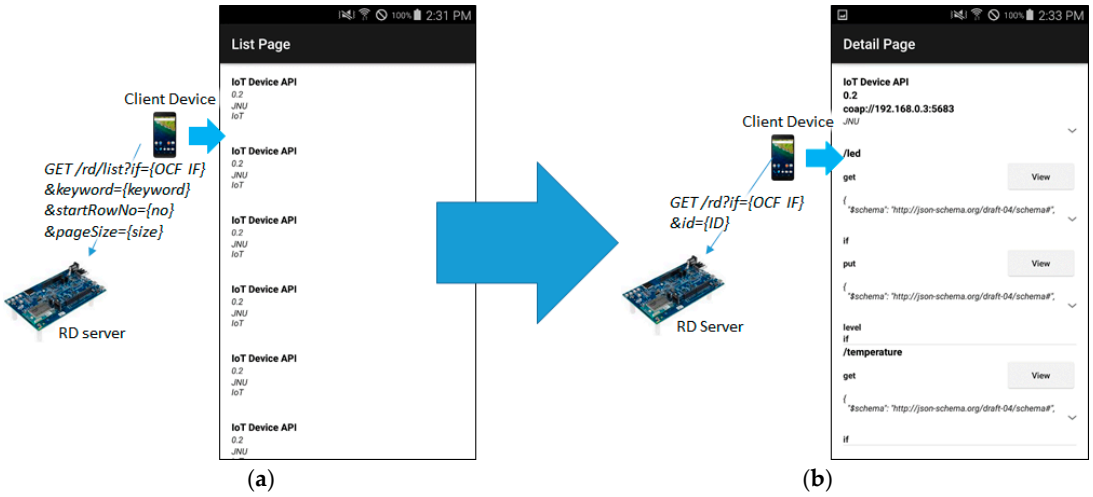

Figure 14. Result of displaying service provider information. (**a**) List page; (**b**) detail page.

Figure 15 shows the results of accessing the IoT services which are discovered by the IoT client. The page in the Figure 15a shows the resource information with the form for sending the OCF request message to the/humidity resource using the GET method. The IoT client can recognize that the request is sent to an IoT device or an HTTP service provider through the information that is registered. If the request is used for requesting an IoT device, the parameters rt and if are used for generating the OCF request. If there are other parameters, then those parameters shall be used for the query parameter of the request. The result is shown in the screen using JSON format. The page in the Figure 15b shows the HTTP service provider service accessing the result through the service page. The page shows the resource information with the form for sending the OCF request message to the/weather resource using the GET method. To access the/weather resource, the parameters q and application identifier (APPID) are required. Once the parameters are filled, and then click the button REQUEST. The IoT client sends the request to the RD server. The RD server forwards the request to the HTTP service provider in the Internet.

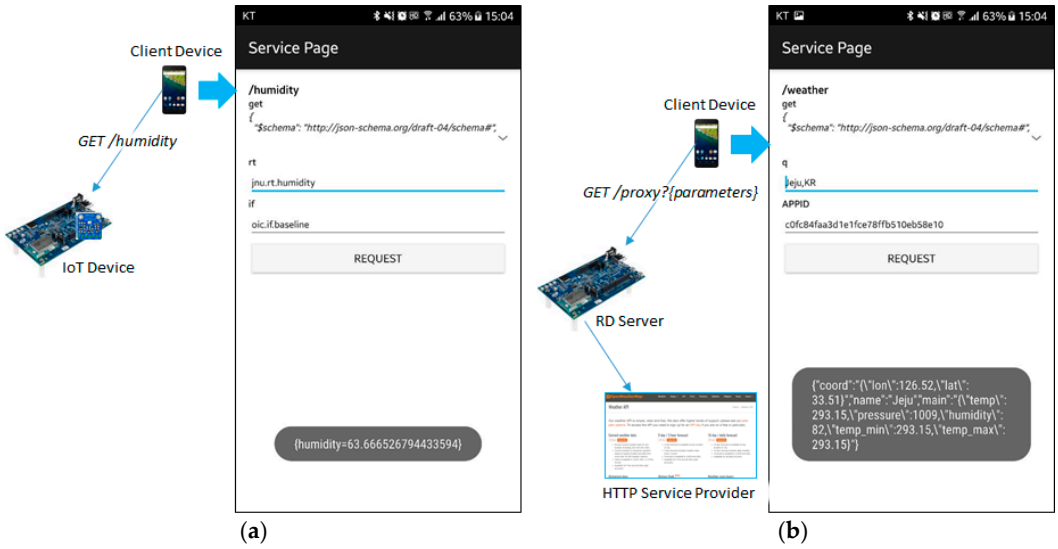

Figure 15. Results of accessing services. (**a**) IoT device service accessing result. (**b**) HTTP service provider service accessing result.

6. Performance Evaluation

The performance evaluations for registration of the IoT device, registration of the publishing client and discovery by the IoT client are presented. Each evaluation is tested by 20 times for the round-trip time (RTT) and the deviation is very small according to the average.

Figures 16 and 17 shows the evaluation results of the network communication delays for registering the IoT device and WSP. For registering the IoT device, the IoT device sends the RAML data to the RD and the RD returns the acknowledgement to the IoT device. Similarly, for registering the WSP, the publishing client is the client for registering the WSP. Figure 18 shows the evaluation results for the discovery by the IoT client. The IoT client requests to the RD and the RD responds the information of registered entities. The RTTs for each registration is done in the same hardware and network environment. The RTT is collected in the IoT device through the time difference between the time for sending the first request and the time for receiving the last response.

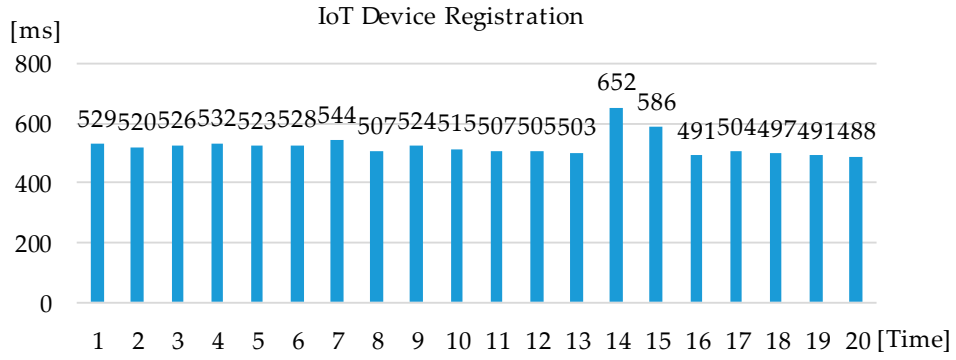

Figure 16. Round-trip time (RTT) for registration by IoT device.

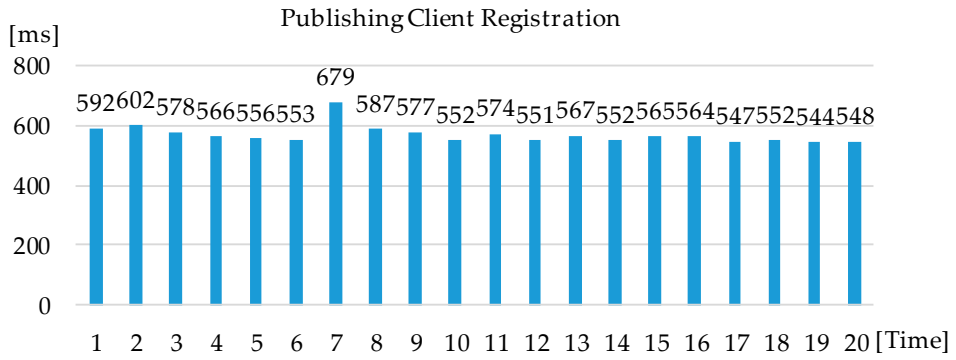

Figure 17. RTT for registration by publishing client.

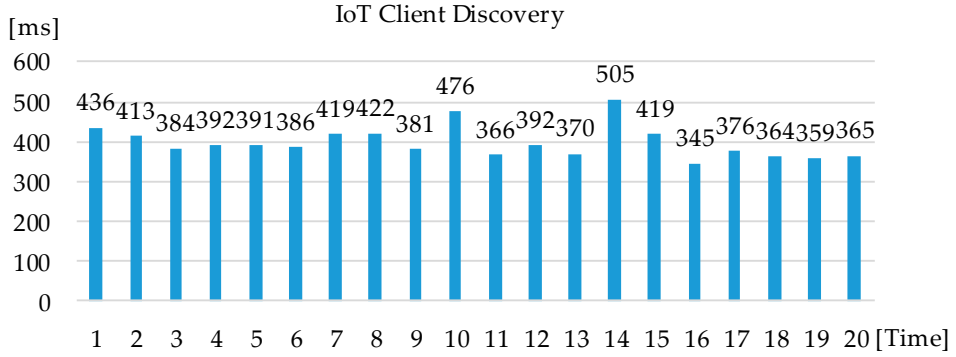

Figure 18. RTT for discovery by IoT client.

The evaluation result for the IoT device registration, that illustrates the RTTs are between 488 ms and 652 ms, the average is 523.6 ms, and the standard deviation is 37.57 ms. The evaluation result for the WSP registration illustrates the RTTs are between 544 ms and 679 ms, the average is 570.3 ms, and the standard deviation is 30.26 ms. The evaluation result for the IoT client discovery illustrates the RTTs are between 345 ms and 505 ms, the average is 398.05 ms, and the standard deviation is 39.92 ms.

Through the performance evaluation, the results illustrate that the interactions take time. The first reason may be the network where the elements communicate. The communication is supported by the OCF IoTivity which is the communication solution based on the CoAP. The CoAP is novel protocol for the constrained environment. Therefore, the implementation of CoAP may not be as mature as HTTP. The problem may be occurred because of the delay of discovery by the IoT client. For example, in the exhibition, there are many people using the WiFi to build the network. The WiFi speed shall be bad even if the signal is strong. If the client requires the discovered result to be presented quickly, then the OCF based discovery may not be sufficient. We will implement the HTTP based discovery scheme to support stable interaction.

7. Conclusions

In this paper, we present a consistent registration and discovery scheme using RAML to enable the information of an IoT device and WSP through the embedded RD server in the OCF-based IoT network. The registration enables the IoT device and publishing client to register the information of the device and WSP using the RAML-based profile. Through the registration, the IoT device registers the IoT device information to the RD, and the publishing client registers the WSPs to the RD. The registered information is included in the RAML definition that is the profile for describing the resource information. The RAML-based profile in the payload of a registration request that involves the basic information of a service provider, such as IoT device and WSP. The resource is also described for accessing the service by the IoT client. Therefore, the RD server provides the information to the IoT clients using the registered information that is published by IoT device and the publishing client based on the RAML definition. The discovery service enables the users to retrieve the service information using the IoT client. Through the discovered information, the IoT client accesses the services of IoT device and WSP. Therefore, using the embedded RD server in the OCF-based IoT network, the registration and discovery of services from heterogeneous providers are supported by the consistent scheme. Moreover, through the message forwarding service of RD server, the users use the same IoT client to consume the services from the IoT network as well as the Internet.

In the future, we will extend the functionalities to support more communication solutions for enabling transparent access to heterogeneous IoT devices. Currently, the proposed scheme only enables the registration of OCF devices and WSPs from the networks of the OCF over the CoAP and the Internet based on the HTTP. The discovery interface is provided to OCF clients for discovering the registered IoT devices and WSPs. Based on the proposed scheme, we will implement a registration interface for the IoT devices which are deployed in the constrained environment using low-energy based communication solutions such as zigbee, BLE, and OCF IoTivity over BLE. The registered IoT resources shall be represented as HTTP-based virtual resources to appear in the Internet. Therefore, this is a consistent discovery interface to provide clients for discovering IoT resources.

Author Contributions: W.J. and D.K. designed the overall system. W.J. implemented the overall system and performed experiments. W.J. and D.K. wrote this paper.

Acknowledgments: This work was supported by Institute for Information and communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2018-0-01456, AutoMaTa: Autonomous Management framework based on artificial intelligent Technology for adaptive and disposable IoT), and this research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2017-2016-0-00313) supervised by the IITP (Institute for Information & communications Technology Promotion). Any correspondence related to this paper should be addressed to Dohyeun Kim.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gartner. Available online: https://www.gartner.com/newsroom/id/3598917 (accessed on 2 May 2018).
2. Li, S.; Da Xu, L.; Zhao, S. 5G internet of things: A survey. *J. Ind. Inf. Integr.* **2018**, *10*, 1–9. [CrossRef]
3. Akpakwu, G.A.; Silva, B.J.; Hancke, G.P.; Abu-Mahfouz, A.M. A survey on 5G networks for the internet of things: Communication technologies and challenges. *IEEE Access* **2018**, *6*, 3619–3647. [CrossRef]
4. Naik, N. Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In Proceedings of the 2017 IEEE International Systems Engineering Symposium (ISSE), Vienna, Austria, 11–13 October 2017.
5. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376. [CrossRef]
6. Kafle, V.P.; Fukushima, Y.; Martinez-Julia, P.; Harai, H. Scalable Directory Service for IoT Applications. *IEEE Commun. Stand. Mag.* **2017**, *1*, 58–65. [CrossRef]
7. Liu, M.; Leppanen, T.; Harjula, E.; Ou, Z.; Ramalingam, A.; Ylianttila, M.; Ojala, T. Distributed resource directory architecture in Machine-to-Machine communications. In Proceedings of the 2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Lyon, France, 7–9 October 2013.
8. Meshkova, E.; Riihijärvi, J.; Petrova, M.; Mähönen, P. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Comput. Netw.* **2008**, *52*, 2097–2128. [CrossRef]
9. Park, S. OCF: A New Open IoT Consortium. In Proceedings of the 2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA), Taipei, Taiwan, 27–29 March 2017.
10. De, S.; Barnaghi, P.; Bauer, M.; Meissner, S. Service modelling for the Internet of Things. In Proceedings of the 2011 Federated Conference on Computer Science and Information Systems (FedCSIS), Szczecin, Poland, 18–21 September 2011.
11. Deschambault, O.; Gherbi, A.; Légaré, C. Efficient implementation of the MQTT protocol for embedded systems. *J. Inf. Process. Syst.* **2017**, *13*, 26–39.
12. Jin, W.; Kim, D.K. Design and Implementation of e-Health System Based on Semantic Sensor Network Using IETF YANG. *Sensors* **2018**, *18*, 629. [CrossRef]
13. Rahmani, A.M.; Thanigaivelan, N.K.; Gia, T.N.; Granados, J.; Negash, B.; Liljeberg, P.; Tenhunen, H. Smart e-health gateway: Bringing intelligence to internet-of-things based ubiquitous healthcare systems. In Proceedings of the 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, USA, 9–12 January 2015.
14. Bhanumathi, V.; Sangeetha, C.P. A guide for the selection of routing protocols in WBAN for healthcare applications. *Hum.-Centric Comput. Inf. Sci.* **2017**, *7*, 24. [CrossRef]
15. Kim, B. A Distributed Coexistence Mitigation Scheme for IoT-Based Smart Medical Systems. *J. Inf. Process. Syst.* **2017**, *13*, 1602–1612.
16. Zhong, C.L.; Zhu, Z.; Huang, R.G. Study on the IOT architecture and gateway technology. In Proceedings of the 2015 14th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), Jiangsu, China, 18–24 August 2015.
17. Ludovici, A.; Calveras, A. A proxy design to leverage the interconnection of coap wireless sensor networks with web applications. *Sensors* **2015**, *15*, 1217–1244. [CrossRef]
18. Surwase, V. REST API Modeling Languages-A Developer's Perspective. *Int. J. Sci. Technol. Eng.* **2016**, *2*, 634–637.
19. Djamaa, B.; Yachir, A.; Richardson, M. Hybrid CoAP-based resource discovery for the Internet of Things. *J. Ambient. Intell. Humaniz. Comput.* **2017**, *8*, 357–372. [CrossRef]
20. Boukhadra, A.; Benatchba, K.; Balla, A. Efficient distributed discovery and composition of OWL-S process model in P2P systems. *J. Ambient. Intell. Humaniz. Comput.* **2016**, *7*, 187–203. [CrossRef]
21. Djamaa, B.; Richardson, M.; Aouf, N.; Walters, B. Towards efficient distributed service discovery in low-power and lossy networks. *Wirel. Netw.* **2014**, *20*, 2437–2453. [CrossRef]
22. Cirani, S.; Davoli, L.; Ferrari, G.; Léone, R.; Medagliani, P.; Picone, M.; Veltri, L. A scalable and self-configuring architecture for service discovery in the internet of things. *IEEE Internet Things J.* **2014**, *1*, 508–521. [CrossRef]

23. Bormann, C.; Castellani, A.P.; Shelby, Z. Coap: An application protocol for billions of tiny internet nodes. *IEEE Internet Comput.* **2012**, *16*, 62–67. [CrossRef]
24. Shelby, Z.; Bormann, C.; Krco, S. *CoRE Resource Directory*; IETF: Fremont, CA, USA, 2018.
25. Jin, W.; Kim, D. A Sleep-Awake Scheme Based on CoAP for Energy-Efficiency in Internet of Things. *Int. J. Inform. Vis.* **2017**, *1*, 110–114. [CrossRef]
26. Yachir, A.; Amirat, Y.; Chibani, A.; Badache, N. Event-aware framework for dynamic services discovery and selection in the context of ambient intelligence and Internet of Things. *IEEE Trans. Autom. Sci. Eng.* **2016**, *13*, 85–102. [CrossRef]
27. Huh, J.H.; Seo, K. Design and test bed experiments of server operation system using virtualization technology. *Hum.-Centric Comput. Inf. Sci.* **2016**, *6*, 1. [CrossRef]
28. Blundo, C.; Orciuoli, F.; Parente, M. An AmI-based and privacy-preserving shopping mall model. *Hum.-Centric Comput. Inf. Sci.* **2017**, *7*, 26. [CrossRef]
29. Datta, S.K.; Da Costa, R.P.F.; Bonnet, C. Resource discovery in Internet of Things: Current trends and future standardization aspects. In Proceedings of the 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), Milan, Italy, 14–16 December 2015.
30. Datta, S.K.; Bonnet, C. Search engine based resource discovery framework for Internet of Things. In Proceedings of the 2015 IEEE 4th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 27–30 October 2015.
31. Barchetti, U.; Bucciero, A.; De Blasi, M.; Mainetti, L.; Patrono, L. Implementation and testing of an EPCglobal-aware discovery service for item-level traceability. In Proceedings of the 2009 International Conference on Ultra Modern Telecommunications & Workshops, ICUMT'09, St. Petersburg, Russia, 12–14 October 2009.
32. Swetina, J.; Lu, G.; Jacobs, P.; Ennesser, F.; Song, J. Toward a standardized common M2M service layer platform: Introduction to oneM2M. *IEEE Wirel. Commun.* **2014**, *21*, 20–26. [CrossRef]
33. Klauck, R.; Kirsche, M. Bonjour contiki: A case study of a DNS-based discovery service for the internet of things. In Proceedings of the 2012 International Conference on Ad-Hoc Networks and Wireless, Belgrade, Serbia, 9–12 July 2012; Springer: Berlin/Heidelberg, Germany, 2012.
34. Evdokimov, S.; Fabian, B.; Kunz, S.; Schoenemann, N. Comparison of discovery service architectures for the internet of things. In Proceedings of the 2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), Newport Beach, CA, USA, 7–9 June 2010.
35. OCF Specification. Available online: https://openconnectivity.org/specs/OCF_Core_Specification_v1.3.1.pdf (accessed on 2 May 2018).
36. Da Xu, L.; He, W.; Li, S. Internet of things in industries: A survey. *IEEE Trans. Ind. Inform.* **2014**, *10*, 2233–2243.