# Distributed resource directory architecture in Machine-to-Machine communications

Meirong Liu[1], Teemu Leppänen[1], Erkki Harjula[1]
[1]Department of Computer Science and Engineering, University of Oulu, Finland
firstname.lastname@ee.oulu.fi

Zhonghong Ou[2], Archana Ramalingam[1]
[2]Department of Computer Science and Engineering, Aalto University, Finland
zhonghong.ou@aalto.fi

Mika Ylianttila[3] Senior Member *IEEE*, Timo Ojala[1]
[3]Center for Internet Excellence, University of Oulu, Finland
mika.ylianttila@cie.fi

***Abstract*— Machine-to-Machine (M2M) communications emerge to achieve ubiquitous communications among all the networked devices. One challenging problem in M2M applications is to discover the resources provided by the devices efficiently. This challenge arises from two perspectives: (1) a large number of heterogeneous devices co-exist and most of them are constrained devices (e.g., limited processing capability), (2) different communication protocols are utilized to get access to different types of resources provided by the devices (e.g., humidity information provided by a sensor). This paper proposes distributed resource directory architecture for M2M applications, referred to as DRD4M. The DRD4M supports heterogeneous devices using HTTP and CoAP protocols for resource registration and lookup. This enables the interoperability among heterogeneous devices and resource access to constrained devices from disparate network including the Internet. The DRD4M introduces two components: a resource registration component for registering resources and a resource lookup component with caching functionality for handling resource lookup with filtering. A peer-to-peer (P2P) overlay is introduced in DRD4M to connect resource peers to avoid single point of failure. A real-world prototype is implemented and is verified with a demo application. Preliminary performance evaluation in terms of response time of resource lookup is provided.**

***Keywords*— *Machine-to-Machine communication; distributed resource directory; overlay; CoAP***

## I. INTRODUCTION

Machine-to-Machine (M2M) applications aim to enable communications among various embedded-networked devices [1] [2] [3]. It involves a large number of nodes (or devices) combined together to enable specific services, e.g., smart metering, intelligent control and industry automation [1] [2] [3]. Direct discovery of resources provided by these devices is not feasible when nodes are sleeping or having intermittent connection to constrained network [4]. Thus, a resource directory is fundamental. On one hand, most of the devices in M2M applications are constrained devices (e.g., having limited memory, and battery-powered [5]). On the other hand, these constrained devices use different communication protocols (e.g., smartphone use HTTP and wireless sensors use Constrained Application Protocol (CoAP) [6]), which increases the complexity of the resource directory. Centralized resource directory architecture can suffers from single-point-of failure. Therefore, one of the challenges is to design a distributed resource directory for a large number of constrained devices that use different communication protocols.

In this paper, we propose distributed resource directory architecture for M2M (DRD4M). The DRD4M supports heterogeneous devices (e.g., wireless sensors, smartphone) using HTTP and CoAP protocols for resource registration and lookup. A resource lookup component with caching function and a resource registration component are designed in the DRD4M for resource management. A peer-to-peer (P2P) overlay is utilized in the DRD4M to connect resource peers in decentralized way. We implement a prototype of the DRD4M and a simple demonstration for showing feasibility in supporting resource description registration and lookup.

The contributions of the paper are as follows：

(1) The DRD4M supports heterogeneous devices for resource description registration and lookup. This is important because it enables the interoperability among heterogeneous devices deployed in disparate network and also enables accessing and monitoring resources provided embedded devices from the Internet (e.g., using smartphone).

(2) A distributed resource directory architecture, which contains resources peers and light resource peers, is proposed to process resource registration and lookup. The resources peers consist of a proxy component, a resource registration component, a resource lookup component with caching functionality, and P2P overlay protocol component. Light resource peers just have the P2P overlay protocol component. A P2P overlay is introduced to connect resource peers and light resource peers in order to avoid the single point of failure and achieves load-balance.

(3) A prototype of the DRD4M is implemented and its feasibility of providing resource directory for constrained devices with different protocols is verified with a demo. Preliminary evaluation results are also provided.

The remainder of this paper is organized as follows: Section 2 reviews the related work. Section 3 first presents architecture of the DRD4M and then introduces the methods of resource description registration and lookup. Section 4 presents the functionality verification and evaluation on the response time of lookup of resource descriptions. Section 5 provides discussion and future work.

## II. RELATED WORK

More and more studies are carrying on M2M communication in recent years. Some studies worked on enabling M2M communications for mobile devices in the Third Generation Partnership Project (3GPP) [8] [9] [10]. Some studies worked on adopting existing communication technologies (e.g., ZigBee and RFID) to enable communications for smart services [11] [12]. Some studies investigated communication protocols for the constrained devices (CoAP) in M2M applications [13] [14] [15] [16]. Communication delay, energy consumption and overhead of CoAP were studied. European Telecommunications Standards Institute (ETSI) presented the M2M functional architecture standard that consists of Device and Gateway Domain, and a Network domain for guidance [17]. Different from the studies mentioned above, our work focuses on the application-level resource management of constrained devices in M2M applications rather than the design or improvement on communication protocols.

Some efforts were made on smart service provision using M2M communications [3] [18]. For example, smart grid technology was utilized to enable utility providers to wirelessly connect to their grid assets (e.g., circuit breakers). These studies focused on using wireless communication to provide more intelligent and reliable services. The services provided by these devices can be accessed directly and the numbers of devices are large. However, this paper aims at applications that have a large number of constrained devices and direct access to resources provided by these devices is not feasible. Thus, a resource directory for this kind of constrained devices is studied in this paper.

The IETF CoRE Working Group made effort to enable constrained devices being utilized in M2M communications [19]. They presented a CoRE resource directory for constrained devices to register and look up resource descriptions [4]. The methods of discovering resource directory, registering and looking up resource descriptions were proposed. CoAP was used by devices to communicate with resource directory. The draft provides guidance on basic functionality design of a resource directory but no architecture design and implementation were provided. Our work focuses on designing a distributed resource directory and provides a real prototype. Mäenpää et al. [20] presented a rendezvous service for CoAP nodes in Wireless Sensor Networks (WSNs). Their work does not support smart devices (e.g., smartphone) for resource registration and lookup using HTTP protocol. In addition, their work cannot support lookup of a set of resources by using some resource properties as query filtering (e.g., coap://rd.example.com/well-known /core?rt="LightLux"). In contrast, our work aims at supporting heterogeneous devices using both HTTP and CoAP. Our work supports handling lookup that contains query filtering, which is achieved by the resource lookup component in the DRD4M.

Many studies also utilized P2P overlay for resource directory of Web and distributed applications (e.g., [21] [22] [23] [24]) because of the decentralization and self-organization of P2P overlay technology. However, in most of studies, the resource directory was not designed for constrained devices (e.g. wireless sensor with 8 KB RAM [5]). For example, Teranishi [23] presented the PIAX for building wireless sensor overlay. The PIAX sends SOAP messages to register resources. SOAP message may generate heavy communication overhead for constrained embedded devices. Our DRD4M targets at constrained devices and supports CoAP protocol.

## III. ARCHITECTURE OF THE DRD4M

### A. *Overview of the DRD4M architecture*

Architecture of the DRD4M is illustrated in Fig. 1. According to the figure, the DRD4M consists of a number of directory peers and light directory peers that connected in P2P manner. A directory peer handles resource description registration and lookup using caching for these constrained devices because the constrained devices usually are not capable of participating in the P2P overlay. A light directory peer participates in the overlay, stores and looks up resources but does not handle requests from clients (i.e., not parsing HTTP and CoAP requests). The design rationale of light directory peer is that some mobile devices can participate in the overlay to store resource and forward lookup request [25], but running as both HTTP and CoAP servers is not feasible in mobile device solely for this purpose.

Each directory peer comprises three layers. The first layer is a proxy component that handles CoAP and HTTP requests. The second layer consists of a resource registration component and a resource lookup component with caching. The resource registration component registers resource descriptions into the overlay and the resource lookup component with caching functionality handles lookup of resource descriptions from the overlay. The bottom layer is P2P communication protocol layer that runs P2P protocol to connect to other directory peers, provides basic storage and lookup in the overlay. A light directory peer only runs the P2P protocol to store and lookup resources from the overlay.

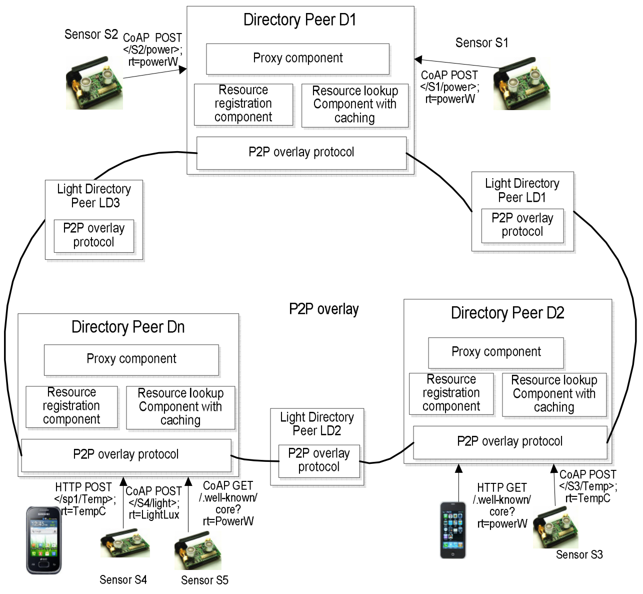

Directory Peer D1
Sensor S2
CoAP POST </S2/power>; rt=powerW
Sensor S1
CoAP POST </S1/power>; rt=powerW
Proxy component
Resource registration component
Resource lookup Component with caching
P2P overlay protocol
Light Directory Peer LD3
P2P overlay protocol
Light Directory Peer LD1
P2P overlay protocol
P2P overlay
Directory Peer Dn
Proxy component
Resource registration component
Resource lookup Component with caching
P2P overlay protocol
Light Directory Peer LD2
P2P overlay protocol
Directory Peer D2
Proxy component
Resource registration component
Resource lookup Component with caching
P2P overlay protocol
HTTP POST </sp1/Temp>; rt=TempC
CoAP POST </S4/light>; rt=LightLux
CoAP GET /.well-known/ core? rt=PowerW
HTTP GET /.well-known/ core? rt=powerW
CoAP POST </S3/Temp>; rt=TempC
Sensor S4
Sensor S5
Sensor S3

Fig. 1. Architecture of the DRD4M.

The DRD4M is designed to support heterogeneous devices and the proxy component provides two handlers for HTTP and CoAP requests. Each handler uses a protocol-specific socket to listen to incoming requests. The handlers parse requests and forward them either to the resource registration component or to the resource lookup component. Meanwhile, the handlers generate and send responses back to requesting devices after getting results from resource registration component or resource lookup component.

## B. *Resource Identification*

One important issue in the design of resource directory is to uniquely identify resource descriptions registered by a device. In the DRD4M, the CoAP URI in a registration request should contain the resource path and endpoint name. For example, in a POST request "coap://rd.example.com/well-known/core/oulu?ep=node1", the resource path is "well-known/core/oulu" and the parameter for the end point name is "ep=node1" [4]. The CoAP URI can be used to uniquely identify registered resources. However, the issue on how to generate the unique endpoint name used in the CoAP URI needs to be resolved. Our DRD4M proposes to generate endpoint name by hashing the MAC address of a node or device. Specifically, the endpoint name of a wireless sensor is generated by hashing its MAC address. The endpoint name of a smartphone is generated by hashing its ANDROID_ID [26].

## C. *Resource Registration*

After discovering resource identification, this section focuses on resource registration mechanism for storing the resource descriptions into the DHT based P2P overlay. Before a node or a device can register its resources into the DRD4M, the node needs to find the address information of the DRD4M. There are a number of ways of discovering the resource directory [4]. The DRD4M adopts the simple way of sending a POST request to a well-known address to get information of the DRD4M.

In the DRD4M, the resource registration component (RGC) is responsible of registering resource descriptions into the P2P overlay when getting parsed requests from proxy component. The P2P overlay protocol layer stores the resource description into the overlay. The mechanism of resource registration works as follows:

(1) The RGC parses the request to retrieve the resource path, endpoint name and IP address of the device who registers the resource description.

(2) Based on the parsed resource identification information in step (1), the RRC looks up in the P2P overlay to find out whether the resource description is stored in the overlay or not. If not stored, the RGC adds the extra resource identification information (endpoint name and IP address) into that original resource description and then stores this edited resource description into the overlay using a key generated by hashing the resource path. The structure of the resource description registered into the overlay is shown in Fig. 3. It includes the IP address, resource path, resource type, content type and endpoint name. The IP address of the device is stored in case that the original address of the message is needed when the registration message is relayed.

Instead of simple hashing the CoAP URI and storing the original payload of the registration message, which was done in Mäenpää et al. [20], the DRD4M stores the extra information: endpoint name and the IP address besides the original registration message. When a client queries for a resource, the client will get the reply message including the IP address metadata. With the IP address information, the client can directly communicate with the device to get the resource value. However, in Mäenpää et al. [20], the client's query is first forwarded to a resource peer that manages the nodes providing requested resource information, and then the resource peer queries that node to get the resource value and sends back the resource value. Only the resource peer knows the address of the nodes providing the requested resources.

(3) The RGC invokes adding resources method provided by the P2P overlay protocol to register the resource description into the overlay, using the key generated in step (2). In addition, each resource path is also registered into the overlay under the root path "/.well-known/core", allowing lookup of all the registered resources and therefore reduces the response time of handling query with filters (e.g., coap://rd.example.com/well-known/core?rt="LightLux"&ep=S0200192).

(4) Finally, the P2P overlay protocol processes the adding resource invocation from the RGC and stores the resource description in the overlay. More information about the P2P overlay protocol can be referred to the work [7].

| IP address | Resource Path | Resource type | Content type | Endpoint Name |
|---|---|---|---|---|

Fig. 2. Structure of a resource description stored in the overlay.

## D. *Resource Description Lookup*

The resource lookup component (RLC) takes care of resource queries and processes lookup results returned by the P2P overlay protocol (if needed). The P2P overlay protocol looks up resource descriptions from the overlay. Caching mechanism is utilized by the RLC to reduce the response time of resource lookup and also enables getting resource value when the devices are in sleeping mode. The RLC caches the most recently queried resource descriptions in the overlay. The procedure of the lookup of resource descriptions is running as follows:

(1) The RLC parses the request to get the resource path (if any) and query filters (if any). Filters are used for query on resources that satisfy some criteria.

(2) The RLC performs one of these operations below (either 2.1 or 2.2) after checking whether filters are included in the query or not.

(2.1) If no filter is included in the query, the RLC generates a key for retrieving the resource description from the overlay. The key generation is the same as that in the mechanism of resource registration. Then, the RLC invokes the lookup method provided by the P2P overlay protocol to retrieve the resource description from the overlay using that generated key. After that, the P2P overlay protocol handles the lookup invocation and returns stored resource description back to the RLC component.

(2.2) If filters are included in the query, the RLC first looks up cached results (stored in the overlay) to find queried resources. If matched results can be found from the cached results, the RLC returns the cached results. Otherwise, the RLC looks up the resources from the overlay and then caches the results with a probability. If some resources are queried more frequently, these resources are assigned high probability to be cached into the overlay, which reduces the time of lookup resource and also enables get resource description even the devices are in sleeping mode or not be able to be accessed.

## IV. PROTOTYPE AND EVALUATION

In this section, we first present the general implementation about the prototype. Then, we introduce the basic functionality test on the prototype with a demo. At last, we evaluate the response time of looking up of resource descriptions.

### A. *Prototype*

A real prototype of the DRD4M is implemented. Chord [27] overlay protocol is used for connecting directory peers and light directory peers. The prototype is deployed in the Gumstix OveroTM Earth board [28]. The constrained wireless sensors are using Atmel 1284P microcontroller-based embedded devices with 8KB of RAM running in 18MHz [5]. The sensor communicates with the DRD4M using CoAP messages over 6LoWPAN. Android smart phone Samsung Galaxy SIII [29] is used for evaluation. The smartphone uses HTTP messages through Wi-Fi network to communicate with the DRD4M and wireless sensors.

### B. *Functionality Verification*

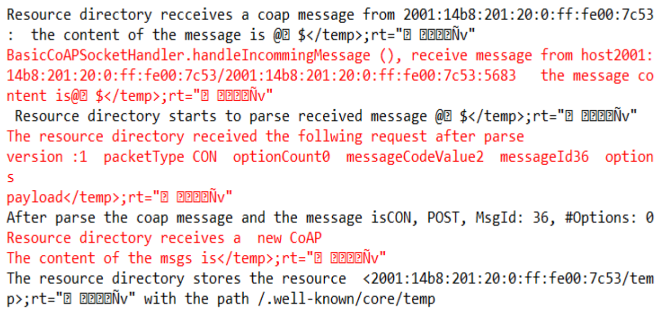
This subsection evaluates functionalities of the DRD4M prototype using a simple demonstration. The DRD4M uses a well-known address. The nodes or devices just send a POST request to get information of the DRD4M.

This demo targets at basic functionality verification to show its feasibility in supporting smartphone to access the resource provided by a sensor. The general idea of the demonstration is as follows: a smartphone provides the resource description of average temperature "avg" and it runs an application of calculating the average temperature in the whole city area. A resource-constrained wireless sensor [5] provides location-based temperature resource (i.e., temp). Firstly, the wireless sensor registers its resource description of temperature (i.e., "temp") into the DRD4M. Then, the smartphone registers its hosted resource "avg" into the DRD4M. When the smartphone travels within the city of Oulu, for example, from city center to university campus Linnanmaa, the smartphone first sends a query to the DRD4M to get the resource description of "temp" in Linnanmaa. Secondly, based on the retrieved IP address from the DRD4M, the smartphone sends another request to get the resource value of "temp" from the wireless sensor. Finally, the smartphone calculates the new average value for the temperature and sends a request to the DRD4M to update the resource description of the "avg" (i.e., adding the new location "Linnanmaa"). The details of the applications running in the smartphone and wireless sensor are out of scope of this paper.

The details of functionality verification are shown below.

(1) A sensor registers a resource "temp" into the DRD4M

Fig. 3 shows that the DRD4M is handing a resource registration message. The DRD4M parses the request and stores the resource description under the path "</.well-known/core/temp>". "/.well-known/core" is a root path used for storing resources if no resource path is specified in the registration request.

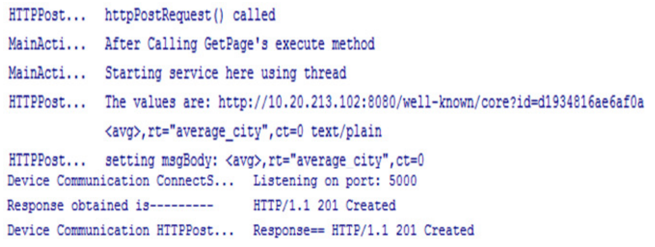


```
Resource directory recceives a coap message from 2001:14b8:201:20:0:ff:fe00:7c53
:  the content of the message is @� $</temp>;rt="� ����Ñv"
BasicCoAPSocketHandler.handleIncommingMessage (), receive message from host2001:
14b8:201:20:0:ff:fe00:7c53/2001:14b8:201:20:0:ff:fe00:7c53:5683   the message co
ntent is@� $</temp>;rt="� ����Ñv"
 Resource directory starts to parse received message @� $</temp>;rt="� ����Ñv"
The resource directory received the follwing request after parse
version :1  packetType CON  optionCount0  messageCodeValue2  messageId36  option
s
payload</temp>;rt="� ����Ñv"
After parse the coap message and the message isCON, POST, MsgId: 36, #Options: 0
Resource directory receives a  new CoAP
The content of the msgs is</temp>;rt="� ����Ñv"
The resource directory stores the resource  <2001:14b8:201:20:0:ff:fe00:7c53/tem
p>;rt="� ����Ñv" with the path /.well-known/core/temp
```

Fig. 3. The DRD4M handles a resource registration sent by the sensor.

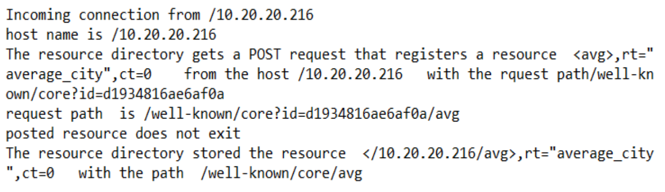
(2) A smartphone registers its hosted resource of "avg" into the DRD4M

Fig. 4 shows the case that the smartphone sends a request to register "avg" into the DRD4M and receives a response of successful creation. Fig. 5 shows that the DRD4M handls a resource registration request sent by smartphone using HTTP and stores the resource description. In the beginning of the evalution, the smartphone is assumed to travel in the city center.



```
HTTPPost...   httpPostRequest() called
MainActi...   After Calling GetPage's execute method
MainActi...   Starting service here using thread
HTTPPost...   The values are: http://10.20.213.102:8080/well-known/core?id=d1934816ae6af0a
              <avg>,rt="average_city",ct=0 text/plain
HTTPPost...   setting msgBody: <avg>,rt="average city",ct=0
Device Communication ConnectS...   Listening on port: 5000
Response obtained is---------      HTTP/1.1 201 Created
Device Communication HTTPPost...   Response== HTTP/1.1 201 Created
```

Fig. 4. The smartphone registers an "avg" resource to the DRD4M.

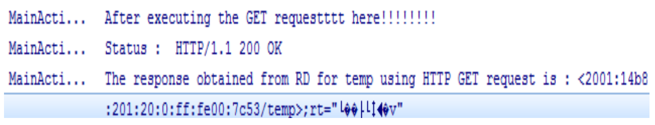


```
Incoming connection from /10.20.20.216
host name is /10.20.20.216
The resource directory gets a POST request that registers a resource  <avg>,rt="
average_city",ct=0    from the host /10.20.20.216   with the rquest path/well-kn
own/core?id=d1934816ae6af0a
request path  is /well-known/core?id=d1934816ae6af0a/avg
posted resource does not exit
The resource directory stored the resource  </10.20.20.216/avg>,rt="average_city
",ct=0   with the path  /well-known/core/avg
```

Fig. 5. The DRD4M handles a resource registration sent by smartphone.

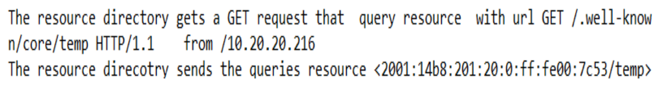
(3) Smartphone queries the resource description of "temp"

When the smartphone travels to university campus Linnanmaa from city center, it calculates the average temperature among the places that it has travelled. As shown in Fig. 6, the smartphone first sends a query to the DRD4M to get the resource description of "temp" in Linnanmaa. Fig. 7 shows the case that the DRD4M handles the query from smartphone and sends back the resource description of "temp" in Linnanmaa.



```
MainActi...   After executing the GET requestttt here!!!!!!!!
MainActi...   Status :  HTTP/1.1 200 OK
MainActi...   The response obtained from RD for temp using HTTP GET request is : <2001:14b8
              :201:20:0:ff:fe00:7c53/temp>;rt="└◆◆├└↕◀◆v"
```

Fig. 6. The smartphone queries “temp” resource from the DRD4M.

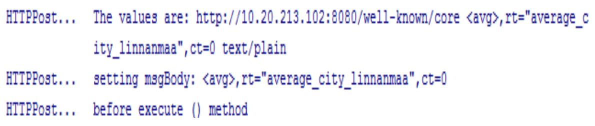


```
The resource directory gets a GET request that  query resource  with url GET /.well-know
n/core/temp HTTP/1.1    from /10.20.20.216
The resource direcotry sends the queries resource <2001:14b8:201:20:0:ff:fe00:7c53/temp>
```

Fig. 7. The DRD4M handles resource query from the smartphone.

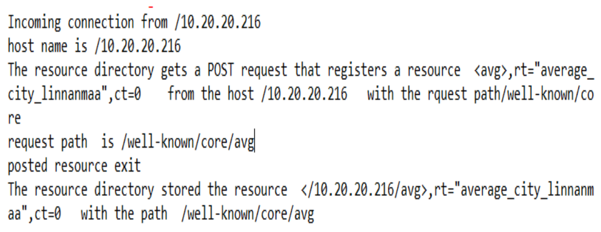
(4) The smartphone updates resource description in DRD4M

Based on retrieved IP address of the sensor in step (3), the smartphone sends another request to get the value of “temp” from that sensor. With the value of “temp” in Linnanmaa, the smartphone calculates the new average value for the resource of “avg”. Finally, the smartphone sends another request to the DRD4M to update resource description of “avg” as shown in Fig. 8. Fig. 9 depicts that DRD4M finds out that the “avg” resource already exits, and updates “avg” by adding semantic information “linnanmaa” into the resource type.



```
HTTPPost...   The values are: http://10.20.213.102:8080/well-known/core <avg>,rt="average_c
              ity_linnanmaa",ct=0 text/plain
HTTPPost...   setting msgBody: <avg>,rt="average_city_linnanmaa",ct=0
HTTPPost...   before execute () method
```

Fig. 8. The smartphone sends a request to the DRD4M to update the resource description of “avg”.



```
Incoming connection from /10.20.20.216
host name is /10.20.20.216
The resource directory gets a POST request that registers a resource  <avg>,rt="average_
city_linnanmaa",ct=0    from the host /10.20.20.216   with the rquest path/well-known/co
re
request path  is /well-known/core/avg
posted resource exit
The resource directory stored the resource  </10.20.20.216/avg>,rt="average_city_linnanm
aa",ct=0   with the path  /well-known/core/avg
```

Fig. 9. The DRD4M handles resource description update.

## C. *Response Time Evaluation*

This subsection presents evaluation on the response time of resource queries (i.e., lookup of resource descriptions). Smartphone is used to send queries to the DRD4M and three types of queries are evaluated.

The first set of evalution is to compare the response time of lookup resource descriptions from the smartphone and that from the DRD4M. The response time has been evaluated for 20 times and the average value is calculated. Results are shown in Table 1, which do not utilize caching for lookup. According to Table 1, when the query includes the information of resource path, it takes much shorter time (21.35ms for the query of “/light?ep=S03001904”) to look up the resource description, compared to long response time for the query without including resource path (81.25ms for the query of “?ep=S03001904”). The reason is that, in the DRD4M, the resource path is hashed to generate the key for storing resource descriptions. If the query contains a filter but does not provide resource path, the DRD4M first needs to look up the stored resource paths under the root path “/.well-known/core” and then searches the resources paths to find the resource descriptions that match the filter. The standard deviation of response time from smartphone is much bigger. For example, when query for “/light?ep=S03001904”, the standard deviation is 31.60. The reason is that the smartphone uses wireless network for communication and the communication delay on sending and receiving messages is affected a lot by the signal quality of wireless network.

TABLE I. RESPONSE TIME FOR RESOURCE QUERY

| **Query** | **Response Time From the DRD4M** | | **Response Time From the Smartphone** | |
|---|---|---|---|---|
| | ***Average time (ms)*** | ***Standard deviation*** | ***Average time (ms)*** | ***Standard deviation*** |
| /light | 20.95 | 7.80 | 30.05 | 15.04 |
| /light?ep=S03001904 | 21.35 | 8.97 | 46.75 | 31.60 |
| ?ep=S03001904 | 81.25 | 22.08 | 122.33 | 46.52 |

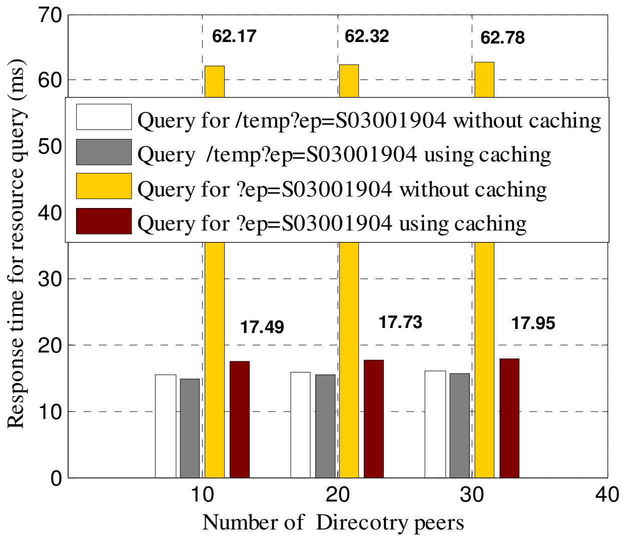


Fig. 10. Response time for resource query using caching.

The second set of evaluation is to evaluate the caching mechanism mentioned in Section III. The response time of lookup of resource description is evaluated when (1) the DRD4M consists of 10 peers, (2) the DRD4M consists of 20 peers and (3) the DRD4M consists of 30 peers. The response time is evaluated for 20 times and the average value is calculated. The response times of lookup of resources with caching and without caching are demonstrated in Fig. 10.

According to Fig. 10, using caching for lookup of resources, the response time for query “?ep=S03001904” is dramatically reduced. For example, the response time is reduced from 62.17 ms to 17.49 ms when the DRD4M consists of 10 nodes. This result shows that the caching functionality benefits a lot for handling the query without including resource path information. Caching benefits only a little bit for handling the query including resource path information. This finding provides important guidance to improve the caching functionality. In addition, the response time only increases a little bit when the nodes increases from 10 to 30.

## V. DISCUSSION AND FUTURE WORK

This paper proposes distributed resource directory for Machine-to-Machine applications (DRD4M). The DRD4M enables the interoperability among heterogeneous devices in disperate networks and enables resource access to low-power resource-constrained embedded devices from the Internet. The DRD4M consists of a proxy component, a resource registration component, a resource lookup component with caching functionality, and the P2P overlay protocol. A real prototype is implemented and preliminary evalution results of response time on lookup of resource descriptions are provided.

The evalution focuses on the response time of lookup of resource descriptions, which is one of the most important functionalities provided by the DRD4M. It is found that caching the most recent query results only benefits a lot for handling queries without including the information of resource path. This finding can be used as a guidance to improve the caching functionality. Another future work is to evaluate the scalability of the DRD4M with a large number of directory peers (e.g., from 1,000 to 10,000). In addition, handling the failure of directory peers should also be investigated.

## ACKNOWLEDGMENT

This work was supported by MAMMOTH project, funded by Finnish Funding Agency for Technology and Innovation (Tekes), Finland. The authors would like to thank Pauli Närhi and Jani Ylioja for setting up the wireless sensor for the functionality test, and Dr. Jiehan Zhou from Center for Internet Excellence for his comments on improving the paper.

## REFERENCES

[1] G. Wu, S. Talwar, K. Johnsson, N. Himayat, and K. D. Johnson, “M2M: from mobile to embedded Internet,” IEEE Commun. Mag., vol. 49, no. 4, pp. 36–43, 2011.

[2] Z. Shelby, “Embedded Web Services,” IEEE Wireless Communications, vol. 17(6) pp. 52-57, 2010.

[3] Q. Hu, Q. Yi, H. Chen, A. Jamalipour, “Recent progress in machine-to-machine communications,” IEEE Communications Magazine vol. 49 (4) pp: 24 – 26, 2011.

[4] Z. Shelby, K. Srdjan, and B. Carsten, "CoRE Resource Directory." http://tools.ietf.org/html/draft-shelby-core-resource-directory-02, 2013. [Expires September 13, 2012] .

[5] T. Leppänen, J. Ylioja, P. Närhi, T. Räty, T. Ojala and J. Riekki, “Holistic Energy Consumption Monitoring in Buildings with IP-based Wireless Sensor Networks”, Proc. Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings, pp. 195-196, 2012.

[6] Z. Shelby, K. Hartke, C. Bormann, and B. Frank, “Constrained Application Protocol (CoAP),” https://datatracker.ietf.org/doc/draft-ietf-core-coap/Internet Engineering Task Force, Work in Progress Internet Draft, 2013 [Expires June 9, 2013].

[7] S. Androutsellis-Theotokis, D.A. Spinellis, “Survey of peer-to-peer content distribution technologies”. ACM Computing Surveys, vol.36(4):335–371, 2004.

[8] S.Y. Lien, K.C. Chen and Y. Lin. “Toward Ubiquitous Massive Accesses in 3GPP Machine to Machine Communications”, IEEE Communications Magazine, vol. 49 (4) pp: 66 – 74, 2011.

[9] K. Ko, M. Kim, K. Bae, D. Sung, J. Kim, J. Ahn, “A Novel Random Access for Fixed-Location Machine-to-Machine Communications in OFDMA Based Systems. IEEE Communications Letters, vol. 16 (9) pp: 1428 – 1431, 2012.

[10] 3GPP TS 36.300 V10.0.0, “Evolved Universal Terrestrial Radio. Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN),” June 2010.

[11] Z.M. Fadlullah, M.M. Fouda, N. Kato, et al. “Toward Intelligent Machine-to-Machine Communications in Smart Grid. IEEE Communications Magazine, vol. 49 (4) pp: 60 – 65, 2011.

[12] O. Bergmann, K.T. Hillmann, S. Gerdes, “A CoAP-Gateway for Smart Homes”, Proc. Computing, Networking and Communications, 2012.

[13] M. Kovatsch, S. Duquennoy, and A. Dunkels, “A Low-Power CoAP for Contiki,” Proc. IEEE conf. Mobile Adhoc and Sensor Systems (MASS), pp. 855–860, 2011.

[14] W. Colitti, K. Steenhaut, N. DeCaro, B. Buta, and V. Dobrota, “Evaluation of Constrained Application Protocol for Wireless Sensor Networks,” Proc. Workshop on Local Metropolitan Area Networks, pp. 1–6, 2011.

[15] K. Kuladinithi, O. Bergmann, T. Ptsch, M. Becker, and C. Gorg, “Implementation of CoAP and its Application in Transport Logistics,” Proc. workshop on Extending the Internet to Low power and Lossy Networks, 2011.

[16] A. P. Castellani, M.Gheda, N. Bui, M. Rossi, and M. Zorzi, “Web Services for the Internet of Things through CoAP and EXI”, Proc. IEEE ICC RWFI Workshop, 2011.

[17] ETSI. Etsi ts 102 690 v1.1.1: Machine-to-machine communications (m2m); functional architecture.

[18] M. Starsinic “System Architecture Challenges in the Home M2M Network,” Proc. Apps. and Tech. NY, 2010.

[19] Constrained RESTful Environments https://datatracker.ietf.org /wg/core/charter/n .

[20] J. Mäenpää, J.J. Bolonio and S. Loreto, “Using RELOAD and CoAP for wide area sensor and actuator networking”, EURASIP Journal on Wireless Communications and Networking , http://dx.doi.org/ 10.1186 /1687-1499, 2012.

[21] M. Balazinska, H. Balakrishnan, D. Karger, “INS/Twine:Scalable Peer-to-Peer Architecture for Intentional Resource Discovery”. Pervasive Computing Lecture Notes in Computer Science, pp:195-210, 2012.

[22] S. Basu, S. Banerjee, P. Sharma, “Nodewiz: Peer-to-Peer Resource Discovery Grids”, Proc. IEEE Cluster Computing and the Grid, 2005.

[23] Y. Teranishi, “PIAX: Toward a Framework for Sensor Overlay Network”. IEEE CCNC, 2009.

[24] T. Koskela, J. Julkunen, J. Korhonen, M. Liu, M. Ylianttila, “Leveraging Collaboration of Peer-to-Peer and Web Services”. Proc Mobile Ubiquitous Computing, Systems, Services and Technologies, 2008.

[25] Z. Ou, E. Harjula, O.Kassinen, M. Ylianttila, “Feasibility evaluation of a communication-oriented P2P system in mobile environments”, Proc. ACM Mobility Conference, 2009.

[26] http://developer.android.com/reference/android/provider/Settings.Secure.html.

[27] I. Stoica, R. Morris, D. Karger, MF. Kaashoek, "Chord: A scalable peer-to-peer lookup service for internet applications", ACM SIGCOMM, 2001.

[28] http://www.gumstix.com/store/product_info.php?products_id=211

[29] http://www.gsmarena.com/samsung_i9300_galaxy_s_iii-4238.php