# Distributed Resource Discovery in the Machine-to-Machine Applications

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

Erkki Harjula[1], Zhonghong Ou[2]
[2]Dept. Computer Science and Engineering, Aalto University, Finland
zhonghong.ou@aalto.fi

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

***Abstract*—One challenging problem in Machine-to-Machine (M2M) applications is to efficiently discover the resources provided by a huge number of heterogeneous devices. This paper proposes distributed resource discovery architecture (DRD) for M2M applications. The DRD supports heterogeneous devices in resource description registration and discovery of resource value. It achieves interoperability among heterogeneous devices in disparate networks and enables resource access from the Internet. In the DRD, a resource registration component is designed for storing resource descriptions; a resource discovery component is designed to retrieve resource values on behalf of clients after getting address information by looking up resource descriptions. The DRD utilizes a peer-to-peer overlay to distribute workload and avoid single point of failure. A real-world prototype is implemented and verified with a simple demo. Preliminary evaluation on response time of resource discovery is provided.**

***Keywords- Machine-to-Machine communication; distributed resource discovery; overlay; CoAP***

## I. INTRODUCTION

Machine-to-Machine (M2M) communication emerges to integrate billons of embedded devices, smart devices to provide control of electricity and water utilities, medical ICT, industry automation (e.g., fleet management) [1] [2]. It has been estimated that there will be 50 billion devices connected to the Internet world-wide by the year 2020 [3]. One challenge in M2M applications is to discover the resources provided by these huge number of heterogeneous devices efficiently. The challenge arises from two perspectives: (1) most devices are constrained devices (e.g., embedded sensor with 8KB RAM [4]), (2) different communication protocols are utilized by these devices (e.g., smartphone use HTTP and wireless sensors use Constrained Application Protocol (CoAP) [5]). In addition, direct discovery of some resources is not feasible when the nodes are sleeping or having intermittent connection in constrained network [6]. Centralized architecture suffers from single-point-of failure and scalability when the number of networked devices grows. Therefore, one solution is to design distributed resource discovery architecture for a large number of constrained devices. IETF Working Group presented a resource directory [6] that defines the methods of looking up resource descriptions instead of resource values.

This paper proposes distributed resource discovery (DRD) architecture to efficiently discover the resource values provided by these constrained devices in decentralized manner. Firstly, the DRD supports heterogeneous devices in resource registration and discovery. This is justified because the DRD supports interoperability among heterogeneous devices in disparate networks and enables access to (or monitoring of) values of resources provided by low-power resource-constrained embedded devices. Secondly, the DRD consists of a proxy component, a resource registration component, a resource discovery component with caching functionality, and Peer-to-Peer (P2P) overlay. The resource discovery component first looks up resource descriptions from the P2P overlay to get address metadata, then sends requests (on behalf of clients) to get the resource values using that address, and finally caches the retrieved value with a probability. (3) A real prototype is implemented and verified with a demo.

## II. ARCHITECTURE OF THE DRD

### A. *Overview of the DRD Architecture*

The DRD Architecture is shown in Fig. 1. As illustrated in the figure, the DRD consists of a number of resource peers that are connected in a P2P overlay. Each resource peer handles resource registration and discovery for constrained devices because the constrained devices usually are not capable of participating in the P2P overlay. Each resource peer consists of three layers. The top layer is a proxy component, which contains two handlers (using protocol-specific sockets) to handle CoAP and HTTP messaging. The middle layer contains a resource registration component that registers resource descriptions into P2P overlay, and a resource discovery component that discovers resource values from the overlay. The bottom layer is the P2P communication layer that runs P2P protocol to provide basic storage and lookup in the overlay.

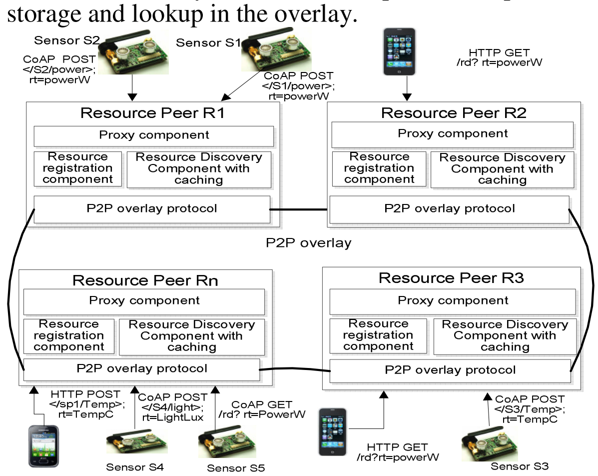


Figure 1. Architecture of the DRD.

### B. *Resource Identification*

One challenge in the DRD design is to uniquely identify resource descriptions. The COAP URI in a request contains the resource path and endpoint name and can be used for resource identification [3]. For example, the COAP URI “rd/oulu?ep=node1” contains the resource path “rd/oulu” and

978-0-7695-5104-3/13 $26.00 © 2013 IEEE
DOI 10.1109/MASS.2013.46

the endpoint name “ep=node1” [3]. However, the issue on generating a unique endpoint name needs to be resolved. The DRD proposes to hash the MAC address of a device (which is unique) to generate the endpoint name. A wireless sensor hashes its MAC address and smartphone hashes its ANDROID_ID [7] to generate endpoint names.

### C. *Resource Description Registration*

Before a device registers resource descriptions into the DRD, it needs to discover the DRD’s address. There are a number of ways to discover the DRD [3]. We adopt the simple way of using a well-known multicast address. In the DRD, the resource registration component (RRC) takes care of registering resource descriptions. The procedure of resource registration works as follows: (1) firstly, the RRC retrieves CoAP URI and IP address from the request. (2) The RRC checks whether the resource description is stored or not. If not, the RRC adds the resource identification (endpoint name and IP address) into the resource description. Next, the RRC generates a key by hashing that resource path and invokes adding resources method provided by the P2P overlay to handle the resource registration. Fig. 2 shows structure of a resource description stored in the overlay. (3) Finally, the P2P overlay layer stores the resource description.

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

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

### D. *Resource Discovery*

The resource discovery component (RDC) takes care of the resource discovery. The RDC adopts a simple way of caching the most recently queried resource values to reduce resource discovery time. The detailed mechanism of resource discovery is as follows: (1) Firstly, the RDC retrieves CoAP URI and generates a key by hashing the CoAP URI. (2) Using the key generated in step 1, the RDC first looks up cached results (stored in the overlay) to find queried resource values. If a matched result is found, the RDC returns that cached resource values. Otherwise, the RDC invokes the lookup method provided by the P2P overlay to retrieve the resource description from the overlay and then sends requests (on behalf of clients) to get values of resources using the address information included in that resource description. The RDC caches the results with a probability that is in proportion to query times (i.e., the more frequent being queried, the higher probability being cached).

## III. PROTOTYPE AND EVALUATION

A real-world prototype is implemented. The constrained wireless sensors use Atmel 1284P embedded devices with 8KB RAM [3]. They communicate with the DRD using CoAP over 6LoWPAN. Samsung Galaxy SIII is used in evaluation as the general purpose Internet-connected embedded device, which communicates with the DRD and sensors using HTTP messages through Wi-Fi. Some screenshots of the basic functionalities are shown in Figs. 3 and 4. Fig. 3 shows the case that the DRD handles resource descriptions from a sensor. Fig. 4 shows a case that the DRD discovers the resource value of “temp”. The response time of discovering resources (i.e., lookup of resource descriptions and retrieve resource values) is evaluated and results are shown in Table 1, in which “std” is the standard deviation. The response time is evaluated for 50 times. The standard deviation of the response time without caching shown in

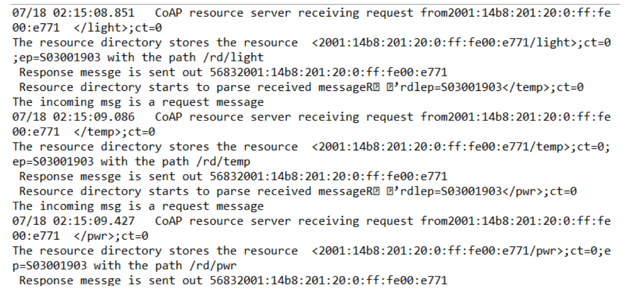


Figure 3. The DRD handles different resource descriptions from a sensor.

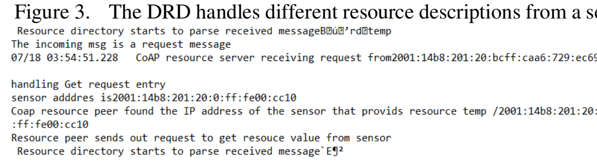


Figure 4. The DRD discovers resource values of “temp” for clients.

TABLE I. THE RESPONE TIME OF DISCOVERING RESOURCE VALUES

| **Operation** | **Average time(ms)** |
|---|---|
| Resource discovery of the DRD without caching | 641.18(std=70.93) |
| Resource discovery of the DRD using caching | 19.13 (std=6.25) |
| The DRD gets resource values from a sensor | 596.92(std=54.14) |

Table 1 is a little bit large. The result of the DRD getting resource values from a sensor shown in the last row of Table 1 also has a large “std”. The reason is that, when the DRD sends requests to the sensor to get resource values, the communication latency is heavily affected by signal quality of the wireless network. When the signal is strong, it takes short time to get resource values from the sensor; otherwise, it takes longer time to get resource values.

## IV. CONCLUSION AND ONGOING WORK

This paper proposes distributed resource discovery architecture (DRD) to discover resources in M2M applications, which achieves interoperability among heterogeneous devices, and enables resource access to resource-constrained embedded devices from the Internet. Ongoing work focuses on deploying the DRD with a large number of nodes and resources to evaluate the scalability, and investigating robustness of the DRD.

## ACKNOWLEDGMENT

The work was supported by MAMMOTH project, funded by Finnish Funding Agency for Technology and Innovation (Tekes), Finland.

## REFERENCES

[1] G. Wu, S. Talwar, K. Johnsson, N. Himayat, and Johnson K. D. “M2M: from mobile to embedded Internet,” *IEEE Commun. Mag.*, vol. 49 (4), pp. 36–43. 2011.

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

[3] CISCO, “The Internet of Things”, http://share.cisco.com/internet-of-things.html

[4] 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.

[5] 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, Internet Draft. 2013,

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

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

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