A Distributed Architecture for Discovery and Access in the Internet of Things

G. Tanganelli, E. Mingozzi, C. Vallati
Dipartimento di Ingegneria dell'Informazione
University of Pisa
I-56122 Pisa, Italy
Email: {g.tanganelli, e.mingozzi, c.vallati}@iet.unipi.it

C. Cicconetti
INTECS S.p.A.
Via U. Forti, 5
I-56121 Pisa, Italy
Email: {claudio.cicconetti}@intecs.it

***Abstract*—A complete, though small-scale, end-to-end architecture for the Internet of Things (IoT) is demonstrated by means of a test-bed including: smart objects (both clients and servers) with limited capabilities and proxy devices which allow the latter to connect to a core network, where a peer-to-peer (P2P) overlay is set up and maintained for automatic discovery of resources and highly scalable access. The Constrained Application Protocol (CoAP) is used by smart objects to enable Machine-to-Machine (M2M) communications following a resource-oriented approach. Business logic and human interactions with the smart objects for aggregation and visualization, respectively, are also demonstrated, including Android mobile applications.**

I. Demo Overview

The Internet of Things (IoT) is rapidly evolving from a market buzzword towards a commercial reality. This is mostly due to the countless applications which, already today, have a high market and social potential and which are viable from a technological point of view. Examples include smart metering infrastructures (e.g., for water, electricity), home automation and smart city applications, personal health systems. Paradoxically, one of the major impediment to the diffusion of the IoT is that its concepts can be applied to many domains, with highly heterogeneous requirements, constraints, and stakeholders, which creates fragmentation and large overlapping of efforts in both industry and academia. While there are some initial projects aiming at unifying standards and systems, e.g., the recently born OneM2M partnership [1], integration and inter-operability remain the two biggest issues to be solved so as to enable a pervasive penetration of the IoT.

In this demonstration we aim at providing a proof-of-concept validation of a complete system made of low-cost and commercial hardware, using standard applications, including Android mobile apps, and protocols, including the Constrained Application Protocol (CoAP) [2], [3]. The system also includes a logical core network for distributed discovery based on peer-to-peer (P2P) which, despite its small size in the test-bed, can be scaled up to realistic numbers for any IoT application domain.

As far as the demonstration is concerned, we highlight that CoAP allows proxying, i.e., it is possible for a node to send the response back to a client on behalf of a remote server. This mechanism can be either transparent to the client (*reverse proxy*) or explicit (*forward proxy*). In the demonstration we make use of reverse proxies to mask the full system to both clients and servers, which are assumed to run on smart objects in an M2M paradigm. More specifically, proxies can retrieve the location of a resource via a peer-to-peer (P2P) overlay based on the eXtendible Metadata Hash Table (XMHT) [7], which is an extension of the widely used Pastry protocol [8]. Each proxy finds its connected servers and crawls through their resources via the CoAP *resource discovery* procedure (see [3] for details). Resources are then advertised to all the other proxies via XMHT.

When a client sends a request to any of the proxies, the selected proxy first finds the server location through P2P querying, then it invokes the requested method on behalf of the client and sends it back the response. Note that resources do not need having unique names (in the context of CoAP the name is the *path* after the address and port in the URI, without the query) in the system, but rather it is a desirable feature that different servers can (unknowingly) host resources with the same name, since this allows replication and semantic aggregation of data.

Finally, we note that our proposed architecture also allows the implementation of the CoAP *Observe* feature to be delegated to proxies: a client requesting to observe a resource will always receive a positive answer from its proxy; in case the remote server does not support the Observe feature, the proxy will implement an active polling towards the latter and inform accordingly the client only when the resource status has changed. This way, network resources in the access network of the client, which are in many cases scarce and hence precious, are preserved.

II. Demo Setup

A logical diagram of the demonstration is shown in Fig. 1, also reporting the network connections between elements.

The demonstration is illustrated by means of the following two use cases. For both use cases, the use of resources is monitored by means of network- and application-level log tools which feed a screen with live data from the system. A proprietary implementation of CoAP is used by the mobile Android applications and the control center system, as well as integrated into the proxies. All the smart objects run the Contiki OS, which embeds CoAP/6LoWPAN/RPL stacks.

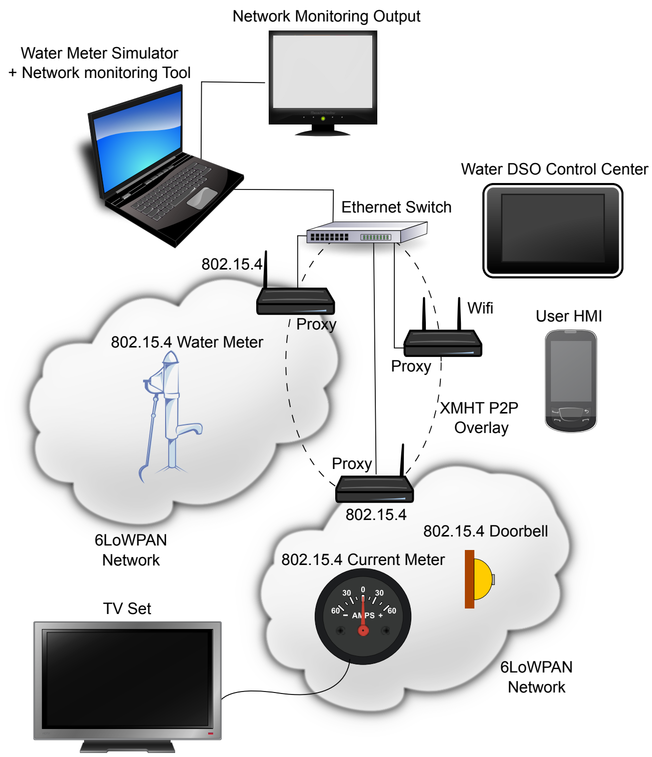

Fig. 1: Demo setup.

A. Use case 1: Smart metering

In this use case the smart objects are metering devices in an Advanced Metering Infrastructure (AMI). In the demonstration a physical sensor is used to measure the amount of water flowing through a pipe in a closed water circuit. Pressure at one end of the pipe is provided by a water pump, which can be enabled/disabled by means of a switch. Additionally, tens of other metering devices are emulated by applications running on a laptop producing data based on pre-computed traces. The artificial data cannot be distinguished from the real data by the other system elements. Each meter is identified by a unique identifier, the name of the zone/neighborhood where it is installed, as well as by its IP address.

In a real deployment, the water Distribution System Operator (DSO) requires a selective acquisition of real time data to, e.g., detect fault or monitor operations during a maintenance mission. An intuitive visual control interface on a tablet is used to emulate the operation center.

Procedure 1: the attendee can select a given identifier by touching the corresponding item on the Human to Machine Interface (HMI). This causes a GET request to be sent to the proxy, which resolves the locator and the method on the server (artificial or real) and sends back the current value to the tablet which visualizes it. When the real sensor is selected, the attendee can shut down or restore the pump, issuing a GET command at any time to see the different reactions with particular regard to the 'separate acknowledgment' mechanism of CoAP, as well as the messages exchanged for the XMHT resolution.

Procedure 2: the attendee is invited to select a given neighborhood from the set of those shown in a separate list on the tablet. This causes a GET request to be sent to the proxy, which resolves the locator and receives back a list of meters residing on the specified neighborhood. A number of GET commands is then issued by the proxy to each meter. The collected results are aggregated and replied back to the client in a single message.

B. Use case 2: Home automation

In this use case the smart objects are a door bell and a current meter connected to a TV set. The door bell hosts a resource whose state can be updated to enable/disable its sound. The current meter hosts a resource whose state represents the value of the current. An user HMI is visualized by the smart phone, where a CoAP client has established an Observe relationship with the current meter sensor. The same graphical interface is also used to update the value of the door bell resource.

Furthermore, the door bell also hosts a CoAP client with an Observe relationship (through the proxy and XMHT overlay) with the current meter, so that when the TV set is switched on (detected based on the current reading exceeding a preconfigured threshold) the door bell is disabled, whereas it becomes enabled again when the TV is switched off. This demonstration shows a simple business logic application implemented in a distributed manner within the network, which exploits the M2M communication enabled by CoAP and XMHT.

ACKNOWLEDGMENT

This work has been partially carried out within the activities of the project "Building the Environment for the Things as a Service (BETaaS)", which is co-funded by the European Commission under the Seventh Framework Programme (grant no. 317674).

REFERENCES

[1] OneM2M partnership. [Online]. Available: http://www.onem2m.org/

[2] C. Bormann, A. Castellani, and Z. Shelby, "Coap: An application protocol for billions of tiny internet nodes," *Internet Computing, IEEE*, vol. 16, no. 2, pp. 62 –67, march-april 2012.

[3] Z. Shelby, K. Hartke, C. Bormann, and B. Frank. (2012) Constrained application protocol (coap). Internet draft. [Online]. Available: http://www.ietf.org/id/draft-ietf-core-coap-12.txt

[4] J. Schneider and T. Kamiya. (2008) Efficient xml interchange (exi) format 1.0. W3C Working Draft. [Online]. Available: http://www.w3.org/TR/exi/

[5] D. Crockford, "Json: The fat-free alternative to xml," in *Proc. of XML 2006*, Boston, USA, dec 2006. [Online]. Available: http://www.json.org/fatfree.html

[6] K. Hartke. (2012) Observing resources in coap. Internet draft. [Online]. Available: http://www.ietf.org/id/draft-ietf-core-observe-07.txt

[7] F. Andreini, F. Crisciani, C. Cicconetti, and R. Mambrini, "Context-aware location in the internet of things," in *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*, dec. 2010, pp. 300 –304.

[8] D. Rowstron, "Pastry Scalable, decentralized object location and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, 2001.