Uniform Web of Things based Access to Distributed Energy Resources via Metadata Registry

UNIFORM WEB OF THINGS BASED ACCESS TO DISTRIBUTED ENERGY RESOURCES VIA METADATA REGISTRY

Aleksei MASHLAKOV
LUT University – Finland
aleksei.mashlakov@lut.fi

Antti KESKI-KOUKKARI
VTT – Finland
antti.keski-koukkari@vtt.fi

Ville TIKKA
LUT University – Finland
ville.tikka@lut.fi

Anna KULMALA
VTT – Finland
anna.kulmala@vtt.fi

Sami REPO
Tampere University – Finland
sami.repo@tuni.fi

Samuli HONKAPURO
LUT University – Finland
samuli.honkapuro@lut.fi

Matti ARO
VTT – Finland
matti.aro@vtt.fi

Peyman JAFARY
Tampere University – Finland
peyman.jafary@tuni.fi

ABSTRACT

A lack of multi-level connectivity between the management systems of smart grid actors and cyber-physical systems of distributed energy resources (DERs) impedes transition toward decentralized grid architecture driven by active network management. This article focuses on Web of Things (WoT) concept as a possible enabler of uniform machine type access to DERs. The results of the paper provide a blueprint of WoT adoption patterns in smart grid domain through the example of microgrid management system (MGMS). Moreover, this article delivers work-in-progress implementation of the metadata registry that facilitates the automated service-oriented discovery of MGMSs by aggregator management systems for purposes of market and grid.

INTRODUCTION

Electric power system is at the rise of fundamental conceptual and structural evolution provoked by rapid penetration of distributed energy resources (DERs). Under scenario of high DER adoption level, this transition could culminate into decentralized smart grid architecture which operation is dynamically driven by multi-party market transactions between cyber-physical systems of customer DERs and centralized management systems of market actors and grid operators [1]. A key operational commodity of such transactions would be DER demand and supply-side flexibility utilized to provide grid stability, network support, market actor optimization, and customer energy security. In order to harness the value of DER flexibility under conditions of high-level modularity of physical grid, tight market coupling, and hardware heterogeneity, an advanced multi-level communication connectivity among the aforementioned systems is critical.

Such connectivity could be attained with Web of Things (WoT) concept that is a further refinement of Internet of Things (IoT) vision and aims to overcome the barriers between the constrained networked environments to enable flexible data exchange between multiple domains for the composition of more complex services and solutions [2]. Web of Things addressed the interoperability issue by establishing a horizontal application-level connectivity between heterogeneous devices and systems through Web technologies. This bridging through open web application programming interfaces (APIs) creates a generic abstraction layer over vertical technological silos and facilitates the interactions and services independent of the underlying standards, communication protocols, and data formats [3]. However WoT vision continues evolving, World Wide Web Consortium (W3C) characterizes an adoption of the core architectural principles of the Web for the WoT as the utilization of semantic annotations based on linked data vocabularies (ontologies) for describing the Things, Unified Resource Identifiers (URIs) for identifying Things and dereference machine-interpretable descriptions of Things, and standard application protocols for accessing the Things [4].

This paper extends WoT vision to the domain of decentralized smart grid to establish a seamless, autonomous, and interoperable environment of technically and economically interconnected systems introduced in HEILA (Integrated Business Platform of Distributed Energy Resources) platform [5]. In particular, this article focuses on microgrid management system (MGMS) presented in [6] and demonstrates its architecture in the context of WoT design patterns. Moreover, the paper describes work-in-progress implementation of metadata registry enabling dynamic integration and unambiguous identification of MGMS in the platform environment as well as automated discovery and uniform access to MGMS by the aggregator management systems (AMSs). The results represent a promising example of plug-and-play market integration of DERs for provision of diverse flexibility services in decentralized smart grid architecture via autonomous machine communications based on Web technologies.

The paper structure is divided into two sections as follows. First section describes MGMS architecture based on the W3C guidance for WoT concept that includes Servient structure, Thing Description (TD), and communication connectivity patterns. Second section is devoted to the implementation of metadata registry where assumptions, registry layered architecture, software components, and security measures are introduces. Finally, conclusion is given and future work is presented at the end of this paper.

MGMS BLUEPRINT FOR WOT

MGMS Servient Connectivity

In WoT the physical Thing is represented by its virtual abstraction called "WoT Servient" that combines the functionalities of both server and client, provides all access and control functions of the web resource, and exposes its interfaces to the network [4]. The building blocks of WoT Servients adopted in HEILA platform for management systems of microgrid resources can be seen from Figure 1. The logic of Servients is embedded in Application Scripts composed of microgrid functionality described in SGAM Functional Layer in [6]. The scripts are executed in an event-loop based state-machine defined in WoT as Runtime Environment. Scripting API is implemented with Smart API [7] that interprets and manages the lifecycle of all the application scripts. Moreover, Smart API also employs client and server APIs bound to HTTPS and MQTT protocols to interact between the platform Servients. Also, system APIs of proprietary and industry protocols are used to communicate with legacy devices.

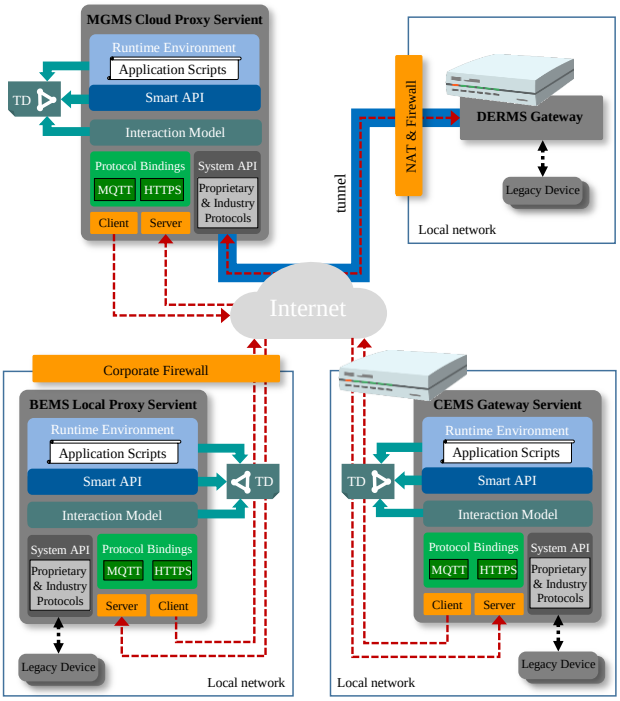

Figure 1. Connectivity patterns for microgrid systems.

Moreover, Figure 1 depicts possible WoT connectivity blueprint for the use case of MGMS orchestrating lower-level DER, customer, and building management systems (DERMS, CEMS, and BEMS respectively) for active network management. In the use case MGMS is defined as Cloud Proxy Servient with a globally reachable address to interact with higher-level AMS and remotely access restricted lower-level systems. CEMS Gateway Servient is introduced as a case for home automation and/or home energy management. This Servient provides to MGMS Servient a universal device accessing method and also wraps various mechanisms for communicating with proprietary devices of consumer electronics. Scenario for DERs owned by an organization is described by BEMS Local Proxy Servient. In this case DERs are usually reside in a local protected network behind a Corporate Firewall. In this configuration BEMS and MGMS Servients act as a combined "Split Proxy" and communicate over secure channels. A final case corresponds to DERs that could be owned by a community such as local PV farm or battery energy storage. The communication with these DERs is built through a secure tunnel (such as an SSH tunnel) between the DERMS Gateway and MGMS endpoint in the cloud using proprietary or industry protocols.

MGMS Thing Description

The basic structure of W3C TD consists of semantic metadata describing the Thing itself, an interaction model based on WoT's Properties, Actions, and Events paradigm, a semantic data schema, security mechanisms, and features for linked data representation [4]. The main idea of TD for MGMS presented in Listing 1 is to highlight the most essential categories that would be enough for AMS to make primarily filtering of microgrids for the required flexibility services and gain the instruction for their access. For this reason, MGMS TD includes aggregated capabilities of microgrid DERs for supported flexibility services, MGMS access interfaces for these services, and location of the microgrid in the network. MGMS TD is semantically described with RDF data model predominately with Smart API ontology which was internally enriched for the platform testing purposes by new entities mostly related to the anticipated organization and terminology of decentralized smart grid architecture. RDF is encoded in Turtle format but can be also converted to JSON-LD or RDF/XML. This TD is not an exhaustive but rather minimum requirement description while more comprehensive data models can be found in [8].

MGMS TD starts with a list of ontologies which URLs are replaced by prefixes in further annotations. It is followed by Green Campus MGMS entity with globally unique identifier URI <lutmicrogrid.fi/mgms>. The URI scheme is omitted since it would misleadingly highlight one of the protocols in MGMS identifier. The locational availability of the physical microgrid within the power network is distinctively described with Electric Connection entity under </connection> path. It is also declared that the MGMS manages Battery Energy Storage, Photovoltaic Panel, and HVAC and offers Conditional Re-Profiling (CRP) service. Maximum parameters for resource availability of CRP service are declared by Resource Flexibility entity under the </resource> path and exemplified for CRP by active power and electric energy. However, in order to identify temporal values of MGMS resource flexibility for CRP service and potentially reserve it, AMS has to utilize MGMS interface capabilities such as flexibility, reserve notification, and verification with necessary access details (topics, ports, etc.). Flexibility

entity corresponds to the Action type in WoT and defines how to request available microgrid flexibility for CRP service from the MGMS via HTTPS. Reserve Notification and Verification Notification entities correspond to the type of Event in WoT. Reserve Notification sequentially follows the Flexibility request and plays the role of transaction for the service booking but it is not described here for the sake of space. Verification Notification is required to verify the provision of booked service. Both notifications are designed with MQTT protocol following request-response pattern through Initialization and Request phases. Initialization requires from AMS to subscribe to the Response topic of MGMS considering that MGMS subscripted to its own Notify topic. AMS activates Request phase by publishing request under MGMS Notify topic that in its turn triggers MGMS to publish requested info under Response topic. Moreover, MGMS TD also adopts Smart API standard paths for the interactions with previously unknown systems. The </discover> path states the standard capability to request the TD directly from the entity while </authorize> path listed here under Data Availability condition specifies the authorization requirements for gaining access to the system data.

```
@prefix ns1: <http://smart-api.io/ontology/1.0/smartapi#> .
@prefix ns2: <http://data.nasa.gov/qudt/owl/qudt#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<microgrid.fi/mgms/access> a ns1:MicroGridManagementSystem ;
   rdfs:label "GreenCampus"^^xsd:string ;
    ns1:hasAvailability </connection> ;
    ns1:hasCapability </discover> ;
    ns1:hasDataAvailability </authorize> ;
    ns1:manages </BESS>, </PV>, </HVAC>;
    ns1:offersService [ a ns1:ConditionalReProfiling;
                        ns1:hasAvailability </resource> ;
                        ns1:hasCapability </flexibility>, </reservenotification>, </verification> ] .

</BESS> a ns1:BatteryEnergyStorage .
</PV> a ns1:PhotovoltaicPanel .
</HVAC> a ns1:HVAC.

</connection> a ns1:ElectricConnection ;
                ns1:valueObject [ a ns1:NationalEnergySystem; rdf:value "Finland" ],
                                [ a ns1:FlexibilityAreaID; rdf:value "FIN1" ],
                                [ a ns1:SecondaryTransformerID; rdf:value 3234 ],
                                [ a ns1:PCC; rdf:value 342133 ].

</resource> a ns1:ResourceFlexibility ;
          [ a ns1:ElectricEnergy; ns2:quantityKind ns1:CapacitiveElectricalEnergy ;
            ns2:unit <http://data.nasa.gov/qudt/owl/unit#Kilowatthour>; rdf:value 100 ],
          [ a <http://data.nasa.gov/qudt/owl/quantity#Power>; ns2:quantityKind ns1:ActivePower ;
            ns2:unit <http://data.nasa.gov/qudt/owl/unit#Kilowatt>; rdf:value 100 ] .

</flexibility> a ns1:Action ;
                ns1:entity [ a ns1:Flexibility ;
                ns1:valueObject [ a <ID> ; ns1:dataType ns1:string ],
                                [ a <Type>; ns1:dataType ns1:string ],
                                [ a <Power> ; ns2:quantityKind ns1:ActivePower; ns2:unit ns1:Watt ],
                                [ a <Start>; ns1:dataType ns1:dateTime ],
                                [ a <Stop> ; ns1:dataType ns1:dateTime ] ] ;

                ns1:interface [ a ns1:InterfaceAddress ;
                                ns1:contentType "text/turtle"^^xsd:string ;
                                ns1:host "lutmicrogrid.fi"^^xsd:string ;
                                ns1:parameter [ a ns1:Parameter; ns1:key "httpMethod"; rdf:value "POST" ] ;
                                ns1:path "/access"^^xsd:string ;
                                ns1:port 8080 ;
                                ns1:scheme "https"^^xsd:string ] ;
                                ns1:method ns1:Read .

</verification> a ns1:Event ;
                ns1:timeSeries [ a ns1:VerificationNotification ;
                                 ns2:quantityKind ns1:ActivePower; ns2:unit ns1:Watt; ns1:list [ ] ;
                                 ns1:temporalContext [ a ns1:TemporalContext ;
                                                       ns1:parameter [ a ns1:Parameter ;
                                                       ns1:key "http://purl.org/dc/terms/hasFormat" ;
                                                       rdf:value xsd:dateTime ] ] ] .

                ns1:interface [ a ns1:Initialize ;
                                ns1:parameter [ a ns1:Parameter; ns1:key "mqttMethod"; rdf:value "subscribe"],
                              [ a ns1:Parameter; ns1:key ns1:topic; rdf:value "microgrid.fi/mgms/access/Response"],
                                [ a ns1:Request ;
                                  ns1:parameter [ a ns1:Parameter; ns1:key "mqttMethod"; rdf:value "publish" ],
                              [ a ns1:Parameter; ns1:key ns1:topic; rdf:value "microgrid.fi/mgms/access/Notify"] ] ;
                                  ns1:method ns1:Read ;
```

Listing 1. MGMS TD described with RDF model.

METADATA REGISTRY

In distributed WoT environment to be visible Servients access Registry Directory that acts as a 'phonebook' to declare their TDs and discover Servients of interest based on the semantic meaning of TDs. The same idea is realized in the proposed metadata registry for autonomous interactions between the MGMSs and AMSs.

Assumptions

An architecture of the metadata registry is based on the assumption of high DER adoption level that would lead to development of distribution level flexibility market platforms, local energy markets and increased amount of self-sufficient energy cells and/or microgrids. Such assumption suggests the division of the registry at the level of MGMS into two layers. A low-level registry corresponds to the internal interactions within MGMS environment while the upper-level is meant for the external interactions between MGMS and management systems of market actors such as AMS. The idea of such registry organization is based on finding the adequate querying discovery mechanism and efficient metadata storage principle considering clustering of the flexibility services for global, regional, and local market perspectives.

Architecture

Upper-level registry architecture presented in Figure 2 was implemented as unilingual multi-database application based on the bottom-up design principles described in [9]. As a point of failure for the autonomous environment it is designed to provide continuous reliability, scalability, and availability even at times of high loads. The architecture has a layered structure divided into Mediator, Wrapper, and Database layers and was built as a multi-container Docker application.

The client request messages come from the IP-infrastructure (Internet) to the registry through NGINX web server which configuration combines the functionality of Reverse Proxy and Load Balancer. Reverse Proxy handles incoming Transport Layer Security (TLS) connections and decrypts the TLS to pass the unencrypted request to the Load Balancer that in its turn distributes the requests to web servers in the Mediator layer. Moreover, a secondary NGINX web server exists that listens heartbeat of the primary to prevent registry failure at the entry point.

The registry Mediator level represents scalable cluster of Web Application Servers, which primary tasks are request distribution among downstream Database Servers and request payload decryption. Request distribution starts with identification of request type between MGMS registration and AMS search requests. The former assumes that the request will be redirected to the Database Server dedicated to the flexibility area stated in MGMS TD while the latter requires investigation about the type of the

service request. It is assumed that all available for the request MGMS flexibility services are divided into global and regional categories that correspond to the System Balancing and Network Management sub use cases of the SGAM Business layer in [7]. If the request is for regional category, then the value of this service has direct dependence on microgrid location, and the request is redirected to the Database Server related to the requested flexibility area. On the other hand, locational dependence is not the case for global services, and in case of this request Web Application Server makes asynchronous requests to all available servers in the Wrapper layer.

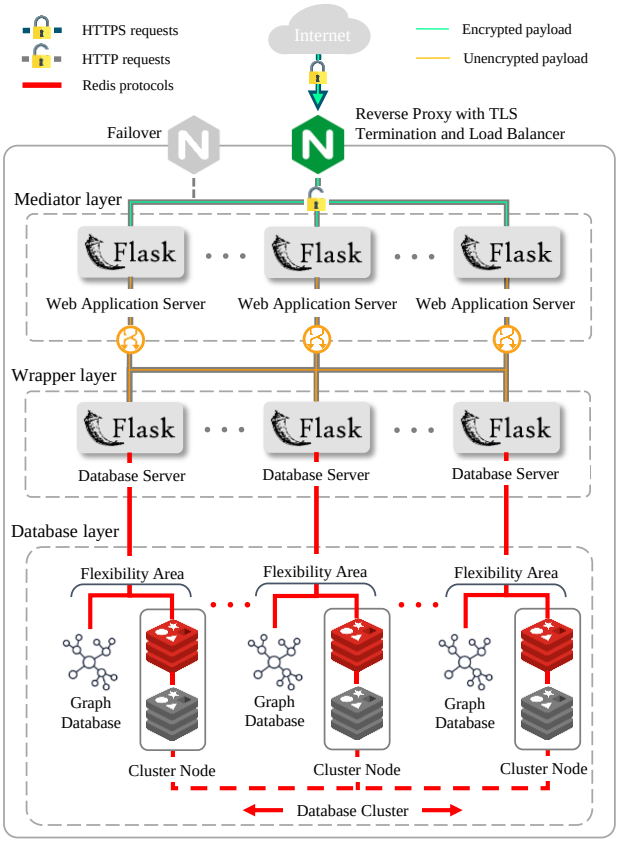

Figure 2. Registry architecture.

The Wrapper layer consists of several Database Servers and performs all the manipulations related to the request transformation from Smart API objects to the data structure appropriate for storing and backwards. The data from the request processed by the Database Server is used to make read and write queries to the Database layer. Both Mediator and Wrapper layers are implemented with Flask micro web framework with configuration supporting multithreading and asynchronous requests.

The Database layer consists of database structures corresponding to dedicated flexibility areas where each structure includes two database types that are Graph and Binary Object Database. The Graph Database is specified to mirror physical medium voltage grid nodes and lines of the corresponding flexibility areas as graph nodes and edges. Labeled property graph structure illustrating this approach is depicted in Figure 3. When the MGMS is registered, it is connected to the graph based on the flexibility area ID, secondary transformer ID and Point of Common Coupling (PCC) ID as a distinguisher between MGMSs connected to the same transformer. It is assumed that such graph organization of the registry would allow to reflect the state of the power grid meaning that every grid reconfiguration would modify the graph structure and predetermine service search results. Described Graph Database is implemented with Redis graph module, and queries to the graph are done with Open Cypher language. The example of AMS query for CRP service is illustrated in Listing 2. Since the data used for MGMS representation and search in the graph do not require most of the data available in TDs, the latter are fully replicated in Binary Object Database which is implemented with high-performance Redis cluster having linear scalability. Every flexibility area has a dedicated master node (marked red in Figure 3) that is responsible of a subset of the cluster hash slots and backed up by one replication node (marked gray in Figure 3). In order to be stored, TD is initially pickled and stored in the Cluster Node based on the MGMS URI that is transformed by hash function to the corresponding hash slot. Furthermore, when URIs of MGMSs satisfying to the AMS search query are identified by the Graph Database, these URIs serve as keys to retrieve MGMS TDs from the Binary Graph Database.

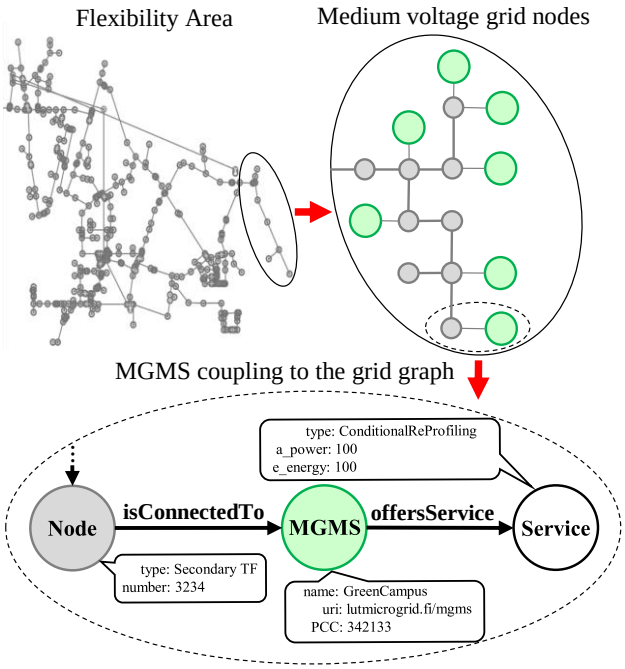

Figure 3. Property graph description.

```
MATCH (n:node)-[:isConnectedTo]->(m:mgms)-[:offersService]->(s:service) \
WHERE s.type='ConditionalReProfiling', s.a_power >=100, n.number =3234 \
RETURN m.uri
```

Listing 2. Open Cypher query for CRP service.

Security

Cyber-security becomes important, and security measures must be applied in order to mitigate the risks and protect data against cyber-security attacks. In the registry implementation TLS provides point-to-point security to the Reverse Proxy by the session keys and TLS record protocol, which guarantee data integrity and confidentiality. In addition, payload encryption adds another security mechanism (end-to-end security) between the external requester and the registry Mediator layer. This payload encryption is implemented as a combination of symmetric and asymmetric key encryption schemes and described in Figure 4 based on the interactions of Registry, MGMS, and AMS. An initial condition is that every MGMS and AMS have their own asymmetric key pairs and possess registry public key (PK). When MGMS is sending the register request, it encrypts the registration payload with the registry PK. Having received MGMS register request, registry decrypts it with own private key and generates symmetric key (SK) for further Registry-MGMS interactions. This SK is sent to the MGMS together with the registration acknowledgement encrypted with MGMS PK. The same approach is used by AMS to interact with the registry using Registry-AMS SK. Moreover, when AMS requests MGMS search, the registry sends response to the AMS consisting of found MGMS TDs and attached to them SK pairs that are further used by AMS to interact with filtered MGMSs. For instance, if MGMS receives request from AMS it cannot decrypt the request payload until it fetches the registry for corresponding MGMS-AMS SK pair. Thus, this mechanism of environment interactions via the registry extends the role of the registry as a trust authority by provision of encryption key management.

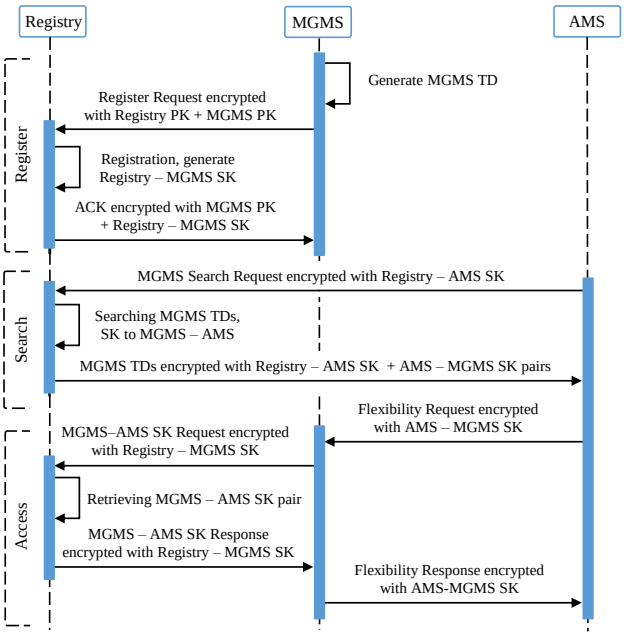

Figure 4. Registry encryption key management.

CONCLUSION

The article provided a blueprint for WoT concept adoption in smart grid domain through the example of MGMS illustrating MGMS Servients connectivity structure and building blocks of and MGMS TD. Moreover, work-in-progress metadata registry implementation was demonstrated as scalable, secure, and reliable solution of MGMS discovery by AMS for diverse market and grid flexibility services in autonomous machine environment. The application of the registry could be extended to MGMS interactions with other smart grid actors to eventually become one of the enablers for machine-driven flexibility market infrastructure.

Future work should investigate how graph theory could be applied to the registry graph database queries to carry out optimization search algorithms for MGMS discovery. Furthermore, including dynamic data about temporal service availability of flexibility resources into MGMS graph description should be researched to find a compromise between pros of this solution for minimizing AMS communication load during search of flexibility services and cons of the environment centralization.

ACKNOWLEDGMENTS

This research was conducted under HEILA project funded by Business Finland, funding decision № 1712/31/2017.

REFERENCES

[1] P. De Martini, L. Kristov, 2015, Distribution Systems in a High Distributed Energy Resources Future: Planning, Market Design, Operation and Oversight. *Future Electric Utility Regulation series. Lawrence Berkeley National Laboratory*.

[2] D. Guinard, V. Trifa, 2009, "Towards the web of things: Web mashups for embedded devices", *Proceedings WWW conference*, vol.15

[3] J. A. Martins, A. Mazayev, and N. Correia, 2017, "Hypermedia APIs for the Web of Things" *IEEE Access*, vol.5, 20058-20067

[4] K. Kajimoto, U. Davuluru, and M. Kovatsch, 2017, "Web of Things Architecture", W3C, W3C Working Draft, [Online]. Available: https://www.w3.org/TR/wot-architecture/

[5] V. Tikka et al, 2018, "Integrated business platform of distributed energy resources - Case Finland", *ICAE*, vol.10

[6] A. Mashlakov, et al. 2018, "Use case description of real-time control of microgrid flexibility." *EEM*, vol.15

[7] 2017, Smart API, Asema Electronics Ltd. [Online]. Available: http://www.smart-api.io/

[8] C. Brunner, A. Schröder, and S. Sucic, 2017, Standardization Punch List, OS4ES, Deliverable 6.5, [Online]. Available: http://os4es.eu/Downloads/

[9] M. T. Özsu, P. Valduriez, 2011, *Principles of distributed database systems*, Springer Science & Business Media.