Autonomous decentralised M2M Application Service Provision

Michael Steinheimer, Ulrich Trick
Research Group for Telecommunication Networks
Frankfurt University of Applied Sciences
Frankfurt/M., Germany
steinheimer@e-technik.org

Michael Steinheimer, Bogdan Ghita
Centre for Security, Communications and Network
Research
University of Plymouth
Plymouth, UK

Woldemar Fuhrmann
Department for Computer Science
University of Applied Sciences Darmstadt
Darmstadt, Germany

Abstract— **In this publication a novel concept for M2M service platform architecture is presented enabling decentralised and autonomous M2M application service provision. This publication introduces the functional architecture of the approach and describes the system structure and application creation process. Additionally several M2M service platforms are evaluated regarding requirements necessary for decentralised M2M application service provision with end user integration. This publication describes also a decentralised M2M application service management process and the corresponding formal notation of M2M application services.**

Keywords— *M2M application service, P2P, service provision*

I. INTRODUCTION

Devices are becoming more and more intelligent by including complex functionality for monitoring and control. Additionally the devices receive the functionality for communication enabling remote monitoring and controlling of devices. Due to the increasing number of intelligent devices, which support forming of intelligent environments, many new application fields can be established (e.g. as presented in [11] smart building, electro mobility, smart city, energy optimisation etc.). Especially the end user domain is an application field with powerful potential, however not or only complicated addressable by external service providers due to e.g. legal restrictions.

Machine-to-Machine Communications (M2M) systems realise the integration of such intelligent devices by provision of specific M2M applications. Traditional M2M systems and the corresponding M2M application services have the focus to support specific business processes. Traditional M2M applications provide their functionality as a service for users (end users or companies). Currently existing concepts from standardisation and research to implement M2M application

services provision (as illustrated in section II) mostly focus on central approaches.

This publication dedicates to present a more flexible methodology for realisation of M2M systems with the focus on dissolving the bindings to centralised entities, integration of the end user for realisation of M2M applications satisfying the individual requirements of end users, and realisation without specialised and dedicated M2M infrastructures. Furthermore, this publication aims to introduce details about the functional architecture of the proposed concept as well as methodologies forming the basis for the presented concept of autonomous decentralised M2M application service provision.

II. M2M SERVICE PLATFORMS

As introduced in section I of this publication the realisation of M2M applications and provision their functionality as a service is typically done using M2M Service Platforms. This section presents requirements for M2M service provision, defined by analysis of existing M2M Service Platform projects. Projects from research area are e.g. INOX Managed Service Platform [4], M2M Platform Project based on SOA [5], BOSP Business Operation Support Platform [6], IMS enabled M2M Service Platform [7], e-DSON [8]. Commercial M2M platforms implement acc. [9] mostly the oneM2M standard for M2M [10]. In order to determine the requirements for M2M service provision with end user environment integration the oneM2M standard as well as projects from research were examined. Table I specifies the analysed projects and an extract of the requirements defined resulting of the strengths and weaknesses of the projects. Disadvantages of the centralised approaches are e.g. that the user of the application is dependent of the platform provider as a central entity. Often M2M platforms provide their functionality only for specific application fields and are therefore less flexible. Centralised

provision of platforms requires many resources (costs for operation, maintenance, availability, application development). End users often have no possibility to define applications themselves (since the creation of applications requires expert knowledge or the end user does not have access to the platform) or the devices present in user's personal environment cannot be connected to the M2M platforms.

The personal environment of the end user described above is difficult to address by external service providers, since the activities (controlling and monitoring of M2M devices) carried out in this area would severely affect the end user or challenge data security. To antagonise this issue, it was proposed in [12] to integrate the end user. For this purpose, an approach was presented in [12-14, 16], which allows end user to define services and automation tasks for their personal environment. The Service Management Framework (SMF) presented in [12-15] consists of a local Service Creation Environment (SCE) and a Service Delivery Platform (SDP). The presented SMF integrates the devices present in the personal area of the end users regardless of their technology and provides basic services, e.g. for multimedia communication. The end user can combine intelligent and communicable devices as well as the basic services using the SCE and thus define personalised services for his personal area (e.g. automation tasks, sensor-dependent signalling or monitoring). An elementary requirement for the platform described above was that it provides an opportunity to develop services intuitively. An intuitive development can be realised by a graphical development process and by modelling the behaviour of a system independently from underlying technology. Both were implemented in the above-mentioned approach. The graphical development process and the underlying methodology are discussed in section III.
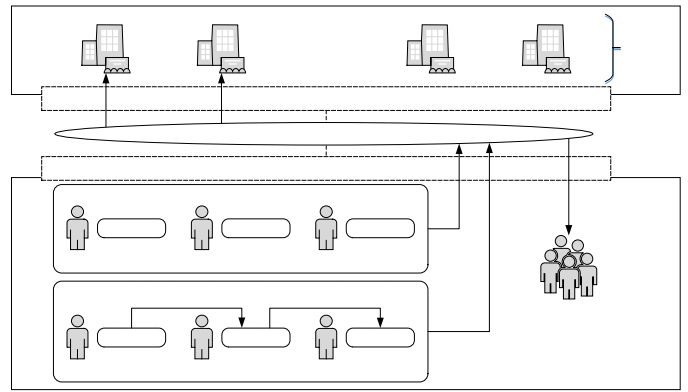

Fig. 1. Cooperative M2M Application Service Provision

By the approach proposed in [11-16], the end user has thus been given the opportunity to create graphically and intuitively services for his personal environment. An approach has been defined in [11, 13, and 16] as to how these services can be made available to other end users or organisations in order to gain added value through the use of the services. In [11, 13, and 16] it was proposed that end users combine the individually offered services to provide a more complex, intelligent and more powerful service. As shown in Fig. 1, a complex service consists of the combination of several distributed services that are networked together. Two ways exist to combine services into an application. Multiple end users can offer the same service with the same local functionality running parallel/ synchronously (service aggregation). Alternatively, end users can offer services that have a different functionality and are composed to an application by linking the services together (service composition).

As described above, the central approaches for defining M2M applications have various disadvantages and are not very flexible. Therefore, as a prerequisite for the approach introduced above, it was defined that creation and provision of individual services and the cooperative provision of an application (consisting of distributed services) is exclusively decentralised, i.e. without the integration of logical as well as physical central entities into the overall system. Under these circumstances, the realisation of the approach cannot be implemented with the existing architectures for M2M platforms. Therefore, an approach has been presented in [11, 13, and 16], which allows implementation following a purely decentralised approach.

III. FUNCTIONAL ARCHITECTURE

Fig. 2 shows the functional architecture of the concept for the autonomous P2P-based provision of M2M application services. The architecture is structured into four layers. The following describes the respective layers in their functionality. **Network Layer:** End user's M2M gateways are connected with each another via an IP network, respectively the Internet. End users can be located at different geographic locations as well as in different access networks (wireless mobile networks or fixed access networks), which are interconnected via a core network. **P2P Network Layer:** End user nodes represent

TABLE I.   EVALUATION RELATED M2M SERVICE PLATFORM PROJECTS

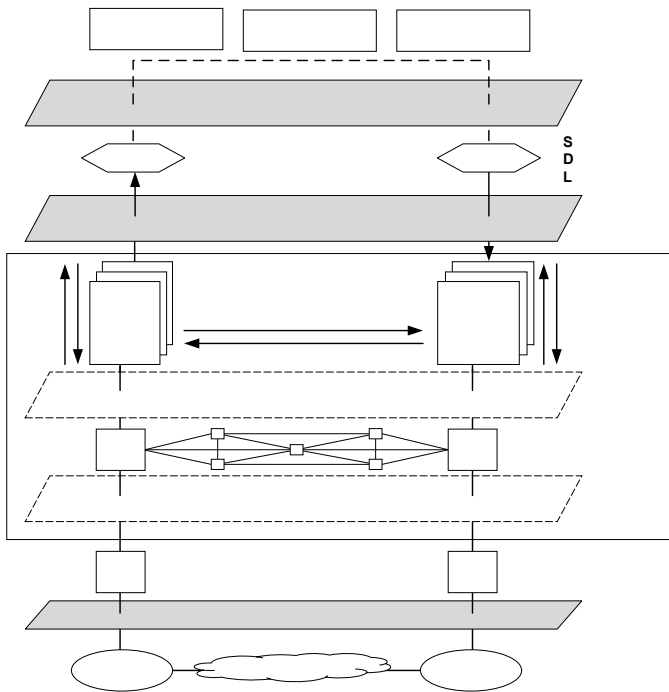| Requirements	M2M Service Platform Projects											
	oneM2M [10]	INOX [4]	M2M on SOA [5]	BOSP [6]	IMS M2M SP I [7]	e-DSON [8]	M2SP [9]	ENERsip [1]	WuKong [2]	COMPOSE [3]	DistribFloodMon [22]	IMS M2M SP II [23]
End user environment integration	-	/	-	+	-	+	+	+	+	+	-	+
End user integration	-	-	-	-	-	o	-	+	+	-	-	-
End user service provision and utilisation	-	-	-	-	-	+	-	-	-	-	-	-
Corporate end user service provision	-	-	-	-	-	o	-	-	-	-	-	-
Decentralised system architecture	o	-	-	-	-	o	-	-	+	-	o	-
Multimedia communication	/	/	-	+	+	-	-	-	-	-	-	+
Device/ service lookup mechanism	+	+	+	+	+	+	+	/	+	+	-	/
Simplicity	-	/	-	-	-	-	-	o	o	o	-	/

Fig. 2. Functional Architecture

equivalent instances and are connected to each other via a P2P network. The P2P Network Layer includes the sub-layer P2P Overlay Layer and P2P Communication Layer. Any data storage in the presented system is decentralised via the end user clients within the P2P overlay layer. For this purpose, the clients, respectively the peers form a P2P overlay network. The overlay network is either a pure structured overlay network or a pure unstructured overlay network (due to the general restriction to avoid also partly centralised system architectures). The overlay network is among other things used as explained in [13-15] to store the addressing information of the peers (assignment temporary to static contact address). In addition, the P2P overlay network is used to manage and configure services and applications (described in section V). The information exchange between the peers for the service utilisation (service requests, confirmations) as well as the necessary signalling to generate the application automatically is also done directly between the peers. For this purpose, various M2M communication protocols (e.g., CoAP, HTTP, SIP, MQTT) can be used. **M2M Service Layer:** The services defined and provided by the end users are available via the M2M service layer and can be used via the corresponding interface. As described in section IV and [12], the services are formally described by a service description language. The service interface is described by an interface specification and is available in the M2M Service Layer (explained in section IV). **M2M Application Layer:** The distributed M2M application services are implemented within the M2M application layer. For this purpose, the services available via the M2M Service Layer are combined with each other as described in section IV. The exchange of information between the services, respectively applications takes place via the P2P network layer by P2P communication protocols. [17]

IV. SYSTEM STRUCTURE AND APPLICATION CREATION

In existing platforms for providing M2M applications, the configuration of automation tasks or the creation of application logic is highly dependent on the implementing systems. To prevent this, the concept defined in this research work was designed with a model-based system structure, derived from [19]. Models, expressed by modelling languages, describe model-based infrastructures and systems. The behaviour of an application or a system is described by means of a platform-independent model, separated from the technology-specific realisation of the system [19]. The behaviour of a system can be modelled with a behavioural model in an abstract manner with much lesser complexity than the platform that implements the behaviour of the system or the application [20]. By stepwise abstraction, respectively transformation, an abstract modelled system is transformed acc. [18] into a concrete system by means of multi-step transformation. Fig. 3 shows the transformation level defined for the approach described in this publication. Derived from [18], transformation and mapping rules have been defined to automatically process the transformation from the abstract model of the system created by the user to the platform-specific models shown in Fig. 3 of the novel architecture for provision of distributed M2M applications described in this publication. As described above, the intuitive development of services and applications is done by graphically modelling the behaviour of a service or creating a behaviour model of service/ application. This allows end users to describe the system without having to have any specific knowledge about the execution platform. The behaviour of a system can be modelled intuitively with a state machine. For this purpose, states represent the devices/ services that should be combined, and transitions represent the connections (information flow) between the devices/ services. To enable end user to model a system using state machine concept, a domain independent modelling language is required which can be used to describe the behaviour of a system abstracted in form of a state machine. [17] The following requirements were initially defined for the selection of an optimal modelling language for this purpose: **Machine readability:** Since no further entities are to be involved in the application generation or its implementation, i.e. the application, respectively the service should be generated automatically after modelling, the modelling language must be machine-readable. It must therefore be a formal language. **Standardised language:** It must be a standardised modelling language in order to ensure the easiest portability. **Immediate mapping of graphical and machine-readable notation:** Since graphical modelling by the end user was defined as a central aspect of the concept, a direct mapping of the graphical notation (elements of the state machine) to the machine-readable notation must exist. **Intuitive usability:** Since the system is to be used by an average technically experienced end user after a short training, the complexity of the graphical notation has to be low. Complex and non-intuitive modelling forms reduce or eliminate end user usability. **Existing parser/ interpreter implementation:** To enable further processing the formal language by a machine, an implementation of a corresponding parser or interpreter must exist. **Nested states:** Combination of application services. A system (M2M application service) should be combined with other systems. A
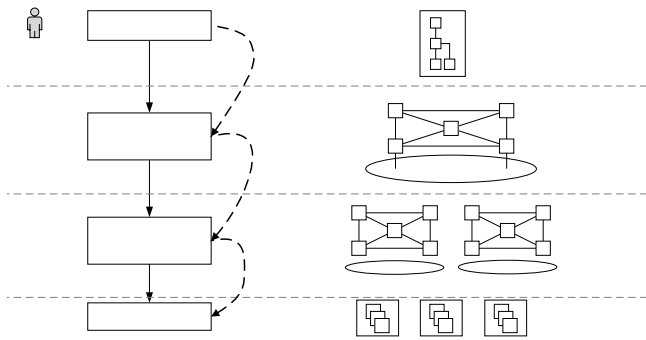
Fig. 3. Transformation Level

previously modelled system is to be reused by being embedded in another application. **Parallel flows:** Within an application, it should be possible to define parallel sequences in order to implement concurrent tasks. **State parametrisation:** The services that are to be combined must have the possibility to be parameterised (definition of configuration/ input and output parameters). Therefore, the states in the state machine must also be parameterisable. **Conditional state transitions:** The transition of a state or transmission of information to a service or the activation of a service should be provided with a condition. **Synchronisation of states:** To synchronise parallel flows, there must be a possibility for synchronisation. According to these requirements, the following state machine-based modelling languages and modelling concepts have been examined: Business Process Modeling Notation (BPMN), Business Process Modeling Language (BPEL), UML State Machines, Petri Nets, and Finite State Machines. The results of the evaluation are shown in TABLE II. The evaluation of the above-mentioned modelling languages showed that UML State Machines fully meet the requirements. UML State Machines are based on Harel Statecharts, which, due to their structure, meet the structural requirements for the state machine. The standardised description language StateChart XML (SCXML) makes it possible to express Harel statecharts in a formal notation. An SCXML parser, which is also available, enables the machine readability and thus the automatic processing of the formally described state machine. [17]

For modelling of a system, the SMF, as described in [12-15], provides a GUI to the end user. Using this GUI, the end user graphically creates a state machine that represents the behaviour of the system. The GUI provides an overview of all available devices and services. The services and smart devices that are in the personal environment of the end users, e.g. in its Smart Home, are stored within the SMF in a local repository. The services offered by other peers are stored in the P2P overlay as described in following section V. An overview of the services stored in the P2P overlay is also loaded by the SCE as described in section V and displayed in the GUI. The end user now graphically combines the available services and devices by first selecting the desired devices or services and then connecting the input and output interfaces defined for the services and devices (transition of the state machine). As described above, the user models the behaviour of service/ application by defining a state machine. The following is a description of the elementary elements of the state machine, as

TABLE II. EVALUATION OF STATE MACHINE MODELLING LANGUAGES

	Modelling Language				
	BPMN	*BPEL*	*UML*	*Petri Nets*	*FSM*
Machine read.	-	+	+	+	-
Standardised	+	+	+	+	-
Immediate map.	-	+	+	+	-
Intuitive usability	+	o	+	-	+
Parser/ Interpreter	-	+	+	+	-
Nested states	o	o	+	+	-
Parallel Flows	+	+	+	+	-
State param.	-	+	+	-	-
Conditional trans.	+	+	+	o	-
Sync. states	o	+	+	+	-

well as the semantic defined by mapping the elements of the state machine to the service/ application components. In addition to these elements, further elements of a state machine can be used, which are not further explained at this point. **State element:** Devices, services, and applications are represented by states. **State parameter:** The state parameters represent the data model of the devices, services and applications. Input, output and configuration parameters can be defined. **Global state machine parameter:** This defines global attributes of a service/ application. Global attributes are available for all services and applications and are queried directly from the source. **Transition:** The transition from one state to another state (transition target) represents the connection of devices, services and applications and the associated information exchange. **Transition condition:** Transitions can be provided with conditions. This defines the precondition when device, service or application sends an information. Conditions can be optionally defined. **Transition assign element:** The transitions of a state machine have assign elements. These assign elements define how the output parameters are assigned to the input parameters of a device, service or application. Thus, the contents of the messages that are exchanged are specified via this element. **Nested state element:** Using the nested state element enables embedding an already defined state machine, i.e. a service or an application, into another service or application. For this purpose, the corresponding state machine is embedded into a nested state element. The nested state element then is integrated as a simple state into the sequence of the state machine. **Parallel state element:** The parallel state element is used to model parallel sequences that can be synchronised as required at the end. [17]

The interface description of a device, service or application represents the template for a state. It is embedded in a service or application description or managed separately. The interface description specifies the input/ output and configuration parameters as well as the value ranges of a service/ application. The interface description is used as a template to represent the devices, services or applications graphically and functionally in the GUI. In addition to the above parameters, the interface description contains further parameters for classifying and describing a device, service or application by e.g. keywords or

prose. The interface description can be stored in a local repository or stored in the overlay network. In addition, it is possible to request the interface description directly from a peer that offers a service. [17]

The SCE automatically generates the formal description of the end user-modelled system using SCXML. Service/ application are completely described by the generated SCXML document. If it is a local service, the application description is stored locally, parsed, interpreted and executed according to the defined conditions. If it is a distributed application, i.e. consisting of distributed services offered by different peers, the application description is distributed to the peers that offer a service in the application context. In this case, each peer receives only the part of the application description that is required to configure and execute its individual service. Peers therefore do not have an overview of the overall system, which performs as a basis for securing the system. [17]

V. APPLICATION REGISTRY, CONFIGURATION, EXECUTION

As described in section III, data storage in the described approach is performed via structured or unstructured P2P overlay networks. The following is a description of the functions for registering and searching, which are implemented using the overlay network for the management of services and applications. Services can be offered by different peers. The peers register their services via the overlay network. To do so, they store the combination of service ID and their associated personal temporary contact information in the overlay. If different peers offer the same services, several associated peers are stored under the same entry in the overlay network. Each peer represents a specific instance of a service. Peers can request the overlay network to get an overview of all available services or a certain subset of all services. This overview is required to generate an application from the various services as described in section IV. Sub-sets of services may include e.g. that services are only available for a particular region, have further meta information that can specify a service more detailed, can only be consumed by a particular group, or implement a particular interface description. Depending on the type of request, the overlay networks offer various ways to provide the overview of the services. For example, the meta-informations stored for a service can be searched by keywords. In order to obtain a specific instance of a service during the application configuration (see below), a peer makes a request (possibly restricted by search criteria) to the overlay network. This returns a list of all found instances of a service. [17]

An M2M application service consists of several services that are interconnected and exchange information. An M2M application is thus a concatenation of distributed services. Each service has input and output interfaces. To realise an application, the output interfaces of a service are connected to the input interfaces of another service, and the information is passed on from one service to the other. The description of the application, i.e., which service is associated with which other service, is defined by a formal application description (described in section IV). The approach described in this publication has two phases to realise an application (illustrated in Fig. 4). A configuration phase in which the application is generated automatically and an execution phase in which the
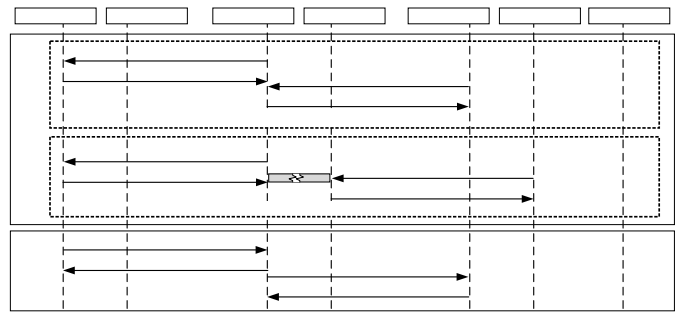

Fig. 4. Application Configuration and Execution Phase

application is running. First, the following section describes the configuration phase. Each peer providing a service involved in an application receives the formal description of the application. As described in section IV, each peer only receives the segment of the application description, which contains the necessary information to configure its individual service and insert it into the overall context of the application according to the application description. Inserting into the overall context of the application means that the peer connects the interfaces (input, output) of his service to the other associated services in order to exchange information. By automatically parsing the application description, the peer determines the following information. **Service connections:** To which other services his offered service should be connected. A distinction is made here between input interfaces served by another service and output interfaces which are served by the service itself. **Input interface information content:** Which information is received via the input interface. **Output interface information content:** What information via the output interface should be transferred to another service. **Output interface condition:** Under which prerequisites an information is to be sent to a service via an output interface (directly on request or time-delayed, if a defined condition is met). **Service configuration parameter:** If configuration parameters have been defined for the offered service, the service must be configured accordingly. **Global attributes:** Information about the attributes provided by services that are not directly connected to their own service via an interface. [17] Each peer now uses the configuration information, determined by the automated parsing of the formal application description, to integrate its service autonomously into the overall context of the application. First, each peer determines specific instances of services to be connected to it, i.e. determines a list of peers that offer the service based on the service ID. This is done as described above. The exchange of information takes place in the approach described in this publication according to the Subscribe/ Notify principle. The connection of an input interface of a service with the output interface of another service is performed by sending the peer with its output interface a subscribe request. If the requested service is available, the requested peer confirms service utilisation request with a positive response. If the service is not available or is available later, the requested peer responds with a negative response and rejects to consume the service or defines a later usage time. Once the connections between the of instances the services are established, the local configuration of the service is performed. This consists of the assignment of the defined input and output information content to the interfaces

(transition assign). Furthermore, the transfer of any existing service configuration parameters to the local service is performed. In addition, the conditions are configured locally that define the preconditions to transmit an information to a connected service via the output interface (transition condition). [17] As described above and shown in Fig. 4, the described approach also includes the execution phase additionally to the configuration phase of an application. In the execution phase, the exchange of information takes place between the peers, which represents the M2M application service. Once the connections between the service instances have been established, the information is transmitted between the services using Notify messages. Sending a notification message either is done directly on request (global attribute or condition less transition) or as soon as a defined condition (transition condition) applies. [17] It should be noted that the same service can be provided by several instances (peers). This may result in incomplete configurations of an application during the configuration phase. In addition to a complete configuration of an application, Fig. 4 also shows an incomplete configuration of an application). For this purpose, an algorithm has been developed which allows all peers to determine independently whether they have embedded themselves in a valid application configuration. The algorithm developed for this purpose is based on the computation of the transitive closure derived from [21] of all connections established in the P2P network in the application context. If a peer detects that it has embedded itself in an invalid application configuration, the established connections are automatically disconnected and, if necessary, a new local configuration process is performed. Thus, it is possible to determine independently a valid application configuration without having to have a complete view of the system. [17]

VI. CONCLUSION

This publication presented principles and details of a novel concept for autonomous and decentralised M2M application service provision. This concept offers new possibilities for applications, realised by several peers, independent of central appliances or corporations. Especially the independent structure of presented system architecture and the simplicity of M2M service/ application creation combined with the autonomous management of M2M application services form the strength of the proposed framework and offer a promising approach for supporting M2M networks with end user environment integration.

REFERENCES

[1] G. López, P. Moura, J. I. Moreno, A. Almeida, "ENERsip: M2M-based platform to enable energy efficiency within energy-positive neighbourhoods", Proceedings of the 2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2011

[2] K. J. Lin, N. Reijers, Y. C. Wang, C. S. Shih, J. Y. Hsu, "Building Smart M2M Applications Using the WuKong Profile Framework", Proceedings of the 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, pp. 1175-1180, IEEE, 2013

[3] C. Doukas, F. Antonelli, "COMPOSE: Building smart & context-aware mobile applications utilizing IoT technologies", Proceedings of the Global Information Infrastructure Symposium (GIIS 2013), 2013

[4] S. Clayman, A. Gali, "INOX: a managed service platform for interconnected smart objects", Proc. of the workshop on Internet of Things and Service Platforms 2011 (loTSP'11), pp. 1-8, December 2011

[5] S. Zhang, J. Zhang, W. Li, "Design of M2M Platform Based on J2EE and SOA", International Conference on E-Business and E-Government pp. 2029–2032 2010 Int. Conf. on E-Business and E-Government, 2010

[6] Q. Xiaocong, Z. Jidong, "Study on the structure of "Internet of Things(IOT)" business operation support platform", 2010 12th IEEE Int. Conf. Commun. Technology (ICCT), pp. 1068-1071, November 2010

[7] L. Foschini, T. Taleb, A. Corradi, D. Bottazzi, "M2M-based metropolitan platform for IMS-enabled road traffic management in IoT", IEEE Commun. Mag., vol. 49, no. 11, pp. 50-57, November 2011

[8] Y. J. Kim, E. K. Kim, B. W. Nam, I. Chong, "Service composition using new DSON platform architecture for M2M service", Information Networking (lCOIN) 2012 Int. Conf. on, pp. 114-119, February 2012

[9] J. Kim, J. Lee, J. Kim, J. Yun, "M2M Service Platforms: Survey Issues and Enabling Technologies", IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 61-76, First Quarter 2014

[10] oneM2M Standardisation Committee, "oneM2M Release 2 specifications",http://www.onem2m.org/technical/published-documents, 2016

[11] M. Steinheimer, U. Trick, W. Fuhrmann, B. Ghita, "P2P based service provisioning in M2M networks", Proc. of Sixth International Conference on Internet Technologies & Applications (ITA 15), Wrexham, UK, 2015

[12] M. Steinheimer, U. Trick, W. Fuhrmann, B. Ghita, "Load reduction in distribution networks through P2P networked energy-community", Proc. of Fifth International Conference on Internet Technologies & Applications (ITA 13), Wrexham, Wales, UK, September 2013

[13] M. Steinheimer, U. Trick, W. Fuhrmann, B. Ghita, "P2P-based community concept for M2M Applications", Proc. of Second International Conference on Future Generation Communication Technologies (FGCT 2013), London, UK, December 2013

[14] M. Steinheimer, U. Trick, P. Ruhrig, R. Tönjes, M. Fischer, D. Hölker, "SIP-basierte P2P-Vernetzung in einer Energie-Community", ITG-Fachbericht 242: Mobilkommunikation, VDE Verlag GmbH, Osnabrück, Germany, ISBN: 978-3-8007-3516-7, 2013

[15] M. Steinheimer, U. Trick, P. Wacht, P. Ruhrig, "Decentralised optimisation solution for Smart Grids using Smart Market aspects and P2P internetworked Energy-Community", IEC/ SGCC/ VDE World Smart Grid Forum 2013, Berlin, Germany, September 2013

[16] M. Steinheimer, U. Trick, W. Fuhrmann, B. Ghita, "P2P-based M2M Community Applications", Proc. of Eleventh International Network Conference (INC 2016), Frankfurt, Germany, July 2016

[17] M. Steinheimer, "Optimierte P2P-Dienstearchitektur für hochverfügbare M2M-Applikationen (P2P4M2M)", TR-004, Frankfurt, Germany, 2016

[18] R. Petrasch, O. Meimberg, "Model Driven Architecture", dpunkt.verlag, Heidelberg, Germany, ISBN: 978-3-89864-343-6, 2006

[19] Object Management Group, "OMG Model Driven Architecture", http://www.omg.org/mda/, January 2017

[20] Object Management Group, "Model Driven Architecture (MDA) MDA Guide rev. 2.0", Boston, USA, June 2014

[21] V. Turau, "Algorithmische Graphentheorie", De Gruyter Oldenbourg, Munich, Germany, ISBN: 978-3-486-59057-9, 2009

[22] S. Kitagami, Y. Miyanishi, Y. Urano, N. Shiratori, "Proposal of a Distributed Cooperative M2M System for Flood Disaster Prevention", Proceedings of the 2014 IEEE 11th Intl Conf on Ubiquitous Intelligence and Computing, 2014

[23] J. E. V. Padilla, J. O. Lee, J. H. Kim, "A M2M horizontal services platform implementation over IP multimedia subsystem (IMS)," Proceedings of the 2013 15th Asia-Pacific Network Operations and Management Symposium (APNOMS), 2013