# M2M Platform with Autonomic Device Management Service

Maroua Meddeb, Mahdi Ben Alaya, Thierry Monteil, A Dhraief, Khalil Drira

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 32 (2014) 1063 – 1070

Procedia Computer Science

International Workshop on Recent Advances on Machine-to-Machine Communication (RAMCOM 2014)

# M2M Platform with Autonomic Device Management Service

M. Meddeb[1,3], M. Ben Alaya[1,2], T. Monteil[1,2], A. Dhraief[3] and K. Drira[1]

[1] *CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France*
[2] *Univ de Toulouse, INSA, F-31400 Toulouse, France*
[3] *HANA Group Lab., Univ de Manouba, Manouba, Tunisie*

**Abstract**

The Internet of Things (IoT) is a way to address and communicate with objects and their virtual representations. The exponential development of Smartphone, tablet, sensors and their applications has created very complex systems. From here, the concept of M2M (Machine-to-Machine) has emerged to provide communication and service architecture without human intervention. The European Telecommunications Standards Institute (ETSI) has proposed a horizontal architecture that can fit to several domains: transport, energy, e-health, etc. A crucial aspect is the management of devices in a system where equipment can be used during decades. Several protocols exist, in this article we explain the OMA DM management protocol, and we describe its connection to the ETSI-M2M platform. An extension of device management is proposed with an autonomic capability.

Selection and Peer-review under responsibility of the Program Chairs.

*Keywords:* M2M, ETSI, Devices Management, OMA DM, autonomic computing;

## 1. Introduction

The Internet of Things has evolved significantly with exceptional speed in the last years. This has allowed connecting a large number of heterogeneous objects (sensors, actuators, Smartphone, application, etc.) through a network such as the Internet. IoT is composed of a series of new independent systems functioning with their proper infrastructures which rest partly on the existing infrastructures of the internet. It covers three types of communications which can be established in restricted (Intranet of objects) or public (Internet of objects) zones: from object to human, from object to object or from machine to machine (M2M).

M2M is the combination of information technology and communication with machines in order to provide them with the mean to interact with minimal human intervention. In the M2M concept, a business application exchanges

* Corresponding author. Tel.: +2-162-150-9871.
*E-mail address:* marouameddeb@gmail.com


Selection and Peer-review under responsibility of the Program Chairs.
doi:10.1016/j.procs.2014.05.534

(bidirectionally) informations with a device through a communication network. The device or/and application can thus act on the basis of the exchanged information. The communication network can be seen in two different aspects; a direct connectivity in which a group of similar devices can interact directly with a single application, or a connection via an M2M Gateway, this case is applicable for "low-cost" devices that may not directly interact with an application because of their limited capacities. The application areas are very large; energy management, home automation, smart metering, eHealth, comfort of living, etc. The initial vision of M2M is to have a multitude of new devices (sensors, actuators, Smartphones, etc.) that can communicate and work together without human intervention in order to increase the comfort and safety of the end user and the economics of energy consumption.

The problem facing the M2M is in the vertical fragmentation of M2M applications currently existing in the different business domain and the lack of interoperability between two different applications belonging to two distinct business domains. Vertical silos shall be substituted by a horizontal platform used by vertical applications. In other words, applications will share a common infrastructure, environments, and network elements. An M2M system described by a clearly structured network, software/hardware interfaces, communication protocols, etc. shall ensure the interoperability of all its elements.
Another problem is the instability of devices state. M2M devices and gateways are installed for long periods of time, and they might run multiple applications, while some of them are installed well after the deployment of the devices. It is simply not possible to assume that the software and applications they run are stable and include all the functionality needed for the foreseeable future.

To solve these problems the European Telecommunications Standards Institute (ETSI) defined in 2012 a specification of a platform of M2M services based on a RESTful architecture. Several M2M platforms were implemented in accordance with the ETSI standardization such as openMTC [1], interDigital [2], etc. In the ETSI M2M architecture there is a specific service named REM (Remote Entity Management) devoted to the device management function. This service makes possible to maintain the stability of a device thanks to the management of performance and error and many other management functions. With this intention, the ETSI standard was interfaced with a management protocol which is the OMA DM (Open Mobile Alliance Device Management) protocol. This protocol makes possible to manage any type of device remotely. Nevertheless, human can't interact with all the necessary management actions. We propose an extension of OMA DM / ETSI M2M based on autonomic computing to enable devices to manage themselves dynamically.

The goal of this article is to explain the functioning of the management service according to ETSI M2M. We specially detail how to introduce management protocol OMA DM in ETSI M2M and propose an extension based on autonomic computing to drive OMA DM actions. This paper is structured as follows. In Section 2, we first outline the ETSI M2M architecture, especially the service capability REM and then explain the OMA DM protocol functionality. We proceed to explain how the OMA DM protocol works with ETSI M2M in Section 3. We finally give an example of management tree manipulation based on autonomic capabilities in Section 4 and conclude in Section 5.

## 2. Related Work

### *2.1. High-Level System Architecture*

ETSI TC M2M (Technical Committee of M2M) has adopted the following high-level system architecture which allows for a common understanding of the system that is under standardization (see Fig 1 [3]).

To properly accomplish the task, the standard ETSI provides applications with M2M Service Capabilities (SC), whatever in the network, device or gateway. The most important advantage of this architecture is that it provides a system with end-to-end communication. This M2M system architecture, maximizes the use of already deployed access and core networks as well as any other form of local and personal area networks [3].
The M2M high-level system architecture includes the concept of the M2M device domain, and the network and application domain.

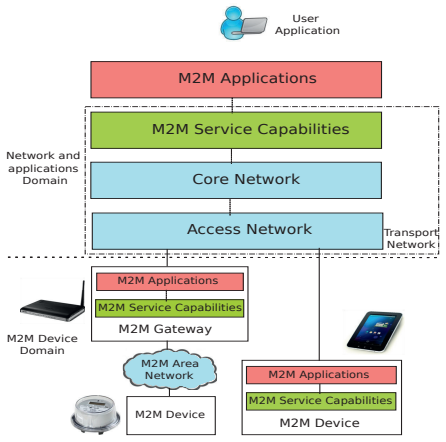


Fig. 1. High-Level System Architecture[3]

### 2.2. ETSI TC M2M Service Capabilities Platform

ETSI proposed a framework that is ultimately a layer of various service capabilities that will be used in M2M communication. The service capabilities layer is represented in the network and applications domain and also on the device and the gateway domain. In addition, ETSI indicates the reference points to allow the exchange of information between services in the same SC layer or services in different SC layers or between applications and services. The reference points constitute initial placeholders for the protocols used to interact between the entities of the M2M system. SCs are extensible; we can add other SCs in the layer and not all SCs proposed by ETSI are mandatory for the functioning of a M2M system[3].

ETSI TC M2M identifies the following SCs for the M2M specification set: x, in each of the following, is N for network, G for gateway and D for device.

- Application Enablement (xAE) provides the single point of contact with applications, all requests sent by applications arrive at this SC.
- Generic Communication (xGC) is the SC through which the communication is made with other services in other domains. It also manages the transport secure and the session establishment.
- Reachability, Addressing, and Repository (xRAR) is the service responsible for storing states of applications, devices and gateways. It also handles subscriptions on data to be notified when it changes. All data are stored in a tree representation called resources tree. Applications, sensors, actuators and gateways are seen as resources that include various details.
- Communication Selection (xCS) occurs in the case where the gateway or the device is accessible via several networks. Therefore, it allows the selection of the network for example WIFI or GPRS.
- Remote Entity Management (xREM) manages devices remotely. The management task is to maintain the life cycle of applications, such as software and firmware upgrade and performance and fault management.
- SECurity (xSEC), through this service the system security is guaranteed. It verifies the identity of different parts, and assures the integrity of the exchanged data. platform.

### 2.3. Remote Entity Management Capability (x REM)

The REM service can perform software and firmware upgrades. In this context software is the operating systems, M2M SCs implemented modules and related APIs. The firmware is a special part of software that controls electrical devices compartments.
In addition, REM is in charge of application life cycle management; it allows to install and uninstall applications or

also to update existing applications.
Another task is Fault and Performance Management (FM, PM), the purpose of this point is to detect errors that reside in different components of the device software or hardware and monitors performance indicators. REM occurs, if possible, in measure to correct these errors or send alerts in the case of degradation of performance to ensure stability of the device's operation. For example, in a device that is battery-operated, it is necessary to monitor the battery level indicator to trigger charging to be sure that the operations of this device will not be interrupted. Other parameters, such as CPU and memory, may be monitored.
The REM service is also responsible for Configuration Management (CM); the REM allows to configure the device parameters to run its proper operations, including the initial configuration.

To perform these management tasks, ETSI M2M proposed to use two management protocols BBF TR069 and OMA DM. M2M Network Applications (NAs) do not have to know the mechanisms of these management protocols or what protocol is used, everything is done in a transparent manner.
ETSI TC M2M defined a resource tree representation which is a meaningful way for addressing resources and to describe how the different types of resources relate to each other. The REM SC acts on a ressouce named "MgmObjs" that is under the node of the managed device. Each management function is represented by an instance of $< mgmtObj >$ under the collection MgmtObjs.

### *2.4. Management Protocol OMA DM*

The OMA DM is a management protocol of mobile devices which makes possible to remotely reach and control devices resources. This protocol is conceived for small devices such as Mobile phones, Smartphones, Tablets, robust laptops, mobile printers, PDAs (Personal DIGITAL Assistant), etc. OMA DM is for sensitive devices, in other words, devices which have a limited storage memory or limited communication bandwidth, like a wireless connectivity[4]. It is difficult for the server to manage mobiles because of their specific configuration and their particular characteristics which depend on each manufacturer. The OMA DM protocol proposed a standard framework making it possible for the suppliers to define the description of the devices and to communicate it to the server. The server is based on this description to send operations. This description is called the management tree of the device. This tree organizes all the managed objects of the device in which each node is addressed (in a single way) by an URI (Universal Resource Identifier). The server is so able, if he has the access right, to modify the management tree by sending operations Get, Add, Delete and Update. The OMA DM protocol consists in exchanging a sequence of XML messages between the server and the client, more particularly the subset defined by SyncML (Synchronization Markup Language). SyncML contains a set of well-defined messages which are transmitted between a server and a client taking part in an operation of synchronization[5]. The communication is carried out in two phases; the phase of initiation then the phase of management of the device[6][7].

### *2.5. Management protocol BBF TR-069*

BBF TR-069 is a protocol for remote management of end-user devices. He offers to the user a set of administration services, control and diagnosis while avoiding the problems involved in material and software diversities. The standard TR-069 was developed for the automatic configuration of the devices with Internet access with the automatic configuration servers ACS (modem, router, Gateway, Set-Top Boxings, VoIP-telephone, etc.).
Since management is done by recovery of the values of devices parameters, the latter are organized according to a hierarchical well-defined structure which is more or less common to all models of devices and manufacturers. This model is published into two formats:

- XML format where each model contains a detailed description of devices as well as the modifications carried out.
- Pdf version container of information readable by human. This last format is seldom used, since it requires a human intervention which is not always practical.

Table 1. Comparative table between OMA DM and BBF TR-069

| | OMA DM | BBF TR-069 |
|---|---|---|
| Data model | Tree | XML file |
| Device type | Mobile Devices | Network Devices |
| Trigger | The server informs Device by SMS or WAP Push | The manager connects to the device embeded http server |
| Architecture Style | Uses the style of architecture REST | Non RESTful management command, RPC |

The manipulation of files is done with a set of operations; “GetParameterNames”, “SetParametersValues”, “GetParametersAttributes”, “AddObject”, “DeleteObject”, etc. Even if the list of the parameters and their attributes are well defined, most devices do not respect the standards completely. The most current problems are the missing parameters. This protocol uses SOAP (Simple Object Access Protocol) to exchange XML messages through RPC (Remote Procedure Call) which enables two applications to do procedure calls one on the other. The communication is carried out in two phases; the phase of initiation then the phase of management of the device [8].

### *2.6. Comparative study between OMA DM and BBF TR-069*

These two protocols are almost similar, we will be able to quote various common points. Both focus on configuration and remote monitoring of devices, the initialization of session is done by the client, the use of SSL (Secure Socket Layer) for the security, the use of XML to exchange messages, the transfer between client and server is done for both management protocols by HTTP protocol, parameters of configuration are presented according to a hierarchical model and finally both handle mutual authentication between server and client before starting the management session.
On the other hand, there are some differences; we list them in the comparative table 1.

Considering that OMA DM protocol uses the tree structure as a data model, and especially the RESTful architecture as in ETSI M2M, it will be more adequate to use it as a management protocol. In addition, the majority of devices do not respect the set of operation proposed by BBF TR-069 while M2M platform aims to be generic.

## 3. Contribution

### *3.1. Integration of OMA DM protocol in M2M ETSI platform*

OMA DM protocol acts on one of the resources in the resources tree to be able to fulfill its functions. This resource is named $< mgmtObj >$. Therefore, it is a question of making a mapping between the two trees; the sub tree of the resource $< mgmtObj >$ of ETSI M2M and the management tree OMA DM.
The strong point of this platform, is that the M2M NA is not supposed to know about the OMA DM protocol mechanism. It just sends a normalized request to receive at the end a normalized answer. Whereas behind this answer, there was a conversion to OMA DM protocol. The request in fact is transformed into OMA DM request and the OMA DM answer is transformed into ETSI format before sending.
Fig 2 provides the architecture that is used to build the ETSI M2M service platform. In this figure, we see two service layers.
The OMA DM client is integrated in the Device or the Gateway service capabilities layer (SCL) in D/GREM service and the OMA DM server is integrated in the Network SCL in the NREM service.

The NA sends an ETSI M2M request to manage a device and this by acting on the resource $< mgmtObj >$ in the resources tree. The request arrives at service NREM and will be mapped in OMA DM format and redirected towards OMA DM server. The OMA DM protocol is a soft protocol, the client is not always listening for requests. So, all requests are stored in the server. Periodically, the client connects to respond to pending requests, this is the initiation phase. Then, the server sends the pending request that will be executed according to the OMA DM protocol in the OMA DM client who resides in the D/GREM. This execution will handle the OMA DM tree of the device to manage it which will generate an answer thereafter. The OMA DM answer will be sent to the OMA DM server to close the

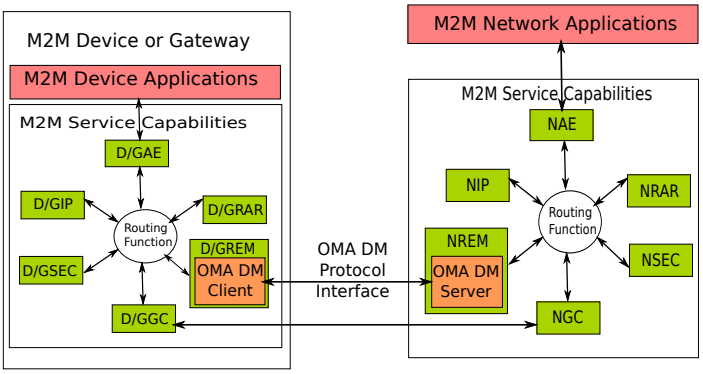


Fig. 2. Integration of device management enablers for M2M REM Service Capability

management session. Once the spot of management carried out, the NREM will execute this update in the same way on the resource < *mgmtObj* > and will end up sending an ETSI answer to the application.

### *3.2. Mapping OMA DM-ETSI trees*

Each node directly under the root in the OMA DM tree represent an instance of the resource < *mgmtObj* > in the ETSI resources tree which provides the same information but which has neither the same name nor the same URI in the tree. So, the most important point in the mapping amounts to replacing the URI of the < *mgmtObj* > resource in the ETSI request by URI in OMA DM management tree of the node to be updated. And it remains to formulate the OMA DM request with the new URI.
The mapping between the resource < *mgmtObj* > and OMA DM tree is done as follows. All nodes directly under the root of the OMA DM tree is mapped to < *mgmtObj* > resources. Sheets directly under these nodes become the attributes of < *mgmtObj* >. The interior nodes under these nodes are < *parmeters* > of the resource father. The sheets of interior nodes are the attributes of the latter[3].
Each resource < *mgmtObj* > in the resources tree has an attribute named "originalMO" which shows the way to attend the equivalent node in the original OMA DM tree. When a request is addressed to a < *mgmtObj* >, we reach his attribute "originalMO" to be able to identify the corresponding node in OMA DM tree[9].

### *3.3. Intelligent management*

With the increasing number of heterogeneous interconnected machines, the traffic of exchanged data became more important, what makes the complexity of M2M systems increases. Providing systems with self-management ability and the permanent assistance is essential[10] . For this purpose, autonomic computing paradigm is introduced to handle complexity. This paradigm was introduced by IBM in 2011 to decrease the high cost of the management of complex system. This approach provides systems with the ability to manage themselves dynamically. Autonomic computing can perform self-configuration, self-healing, self-optimizing and self-protecting[11].
The device management with the OMA DM protocol ensures its normal operation and steady state. The autonomic aspect in the management task is very important to automatically trigger the necessary instructions as soon as we have an alert and not expect a periodic diagnosis that may be delayed.
In AC the device must be aware of his state; with OMA DM protocol it will be easy to monitor and analyse situations through the management tree. The autonomic system collects the status information from OMA DM tree. After analyzing this information, the autonomic system reports the undesirable situation to the NREM ETSI SC. According to the alert, the NREM SC will plan and send a specific management request from the OMA DM server . A session will be established between OMA DM server and OMA DM client in order to manage the device. After the execution of the request in the management tree, the OMA DM client will send a response to report about the status of the performed operations, in particular as to whether this was successful or not. If successful, the NREM will thus make the NRAR SC update the corresponding resource in the resources tree.

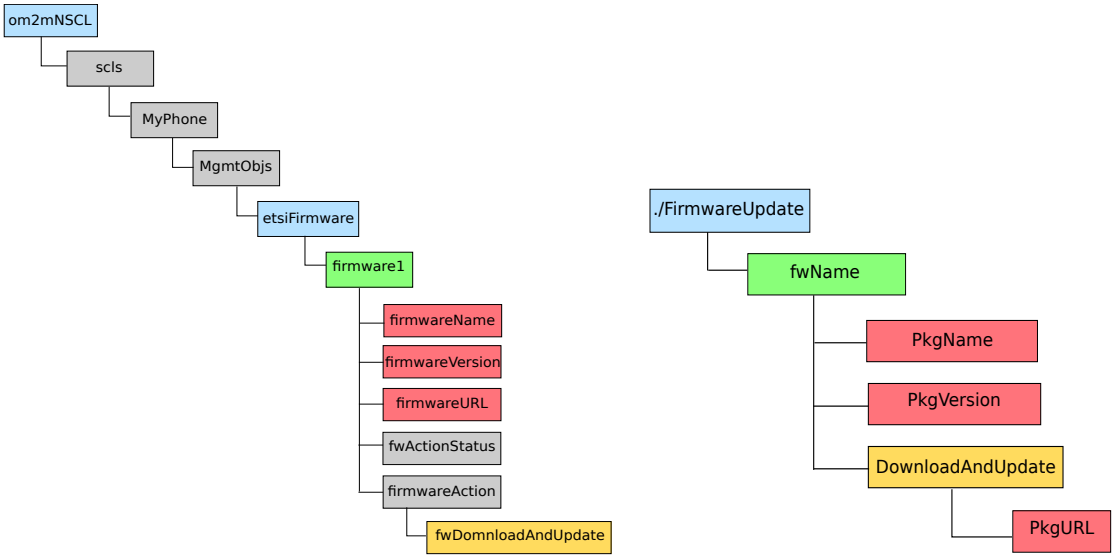


Fig. 3. (a) ETSI MgmtObj subtree; (b) OMA DM Management Tree.

## 4. Example of device management

### *4.1. Simple example*

In order to illustrate the use case for NREM, we provide a scenario showing how NREM can be used for device management. We detail how the NREM service capability performs a firmware upgrade for a device named "MyPhone" at the request of an M2M NA [12].

Initially, the M2M NA requests a firmware update. The parameters for such a request must include an identifier of the device subscription as well as the URL of the firmware image stored on a firmware server.

The $< mgmtObj >$ instance responsible for the firmwares management operation is named "etsiFirmware". For example, the NA requires downloading and updating a new version of a firmware. It will send a set of instructions for that.

First, NA sends a request of creation of this action "fwDownloadAndUpdate" under the resource MgmtObjs of the specific device with this URL: om2mNscl/scls/MyPhone/MgmtObjs/etsiFirmware/firmware1/firmwareAction. "firmware1" is the name known by the application which is not necessarily the true name of the firmware. The original name will be stored in the resource "firmwareName" (Fig 3.a). The request will be mapped in OMA DM format and the node "DownloadAndUpdate", in the same way will be created under the node: ./FirmwareUpdate/fwName. Also "fwName" is not the true name (Fig 3.b).

Then, NA sends a request to update the value of the node "firmwareURL" with an ETSI update request to the URL: om2mNscl/scls/MyPhone/MgmtObjs/etsiFirmware/firmware1/firmwareURL. The new data is the URL from which the device will download the firmware.

In the same way there will be a modification at the URI ./FirmwareUpdate/nomFirmware/DownloadAndUpdate/PkgURL in the OMA DM tree.

Finally, NA sends an Execute request to the URL: om2mNscl/scls/MyPhone/MgmtObjs/etsiFirmware/firmware1/firmwareAction/fwDownloadAndUpdate thus mapped towards URI: ./FirmwareUpdate/nomFirmware/DownloadAndUpdate. When the device receives this request and it is able to satisfy it, it sends an acquittal to the server. At this time, the value of the node om2mNscl/scls/MyPhone/MgmtObjs/etsiFirmware/firmware1/fwActionStatus will be put at "update_progress".

DREM will then download the firmware image, install it, and initiate a reboot. Then DREM establishes a session with the DM server in order to report about the status of the performed operations, in particular as to whether these

were successful or not. If successful a confirmation with the firmware information is sent back to the NA. The NREM will thus update this information as well as the status to "update_successful".

### *4.2. Intelligent example*

Contrary to the first example, in this one, the device asks for a management request. So, with AC, the device doesn't need to wait for the server to begin a management operation.
Thanks to the OMA DM tree the device or the gateway is aware about its state as well as the state of the devices which are connected to it. A device can face problems such as the dysfunction of a software, either it receives a request to execute an operation which cannot handle. In the case of a gateway, it can be connected to a new type of device so it cannot communicate with it, it can also receive information from a device which it can not process.
In these situations, a device or a gateway reports the problem to the OMA DM server to make it starts management actions; updates of the already installed applications or the installation of the new missing applications.

## 5. Conclusion

Vis-a-vis the problem of the heterogeneity of the equipment and technologies in the Internet of Things, ETSI proposed a standard in which it details the specification of a platform of M2M services. In this article, we detailed the platform of services M2M, which covers the features of management of a remote device by using the management protocol OMA DM. We also put the focus on self-management of devices by introducing the Autonomic Computing paradigm.
As a challenge we propose to experiment autonomic device management in a more complex scenario to manage energy consumption in the Smart Metering domain. In addition, we aim to use our approach for the self-configuration of new devices present in the vicinity of the management server.

## References

1. Wahle, S., Magedanz, T., Schulze, F.. The openmtc framework - m2m solutions for smart cities and the internet of things. In: *WOWMOM*. IEEE Computer Society. ISBN 978-1-4673-1238-7; 2012, p. 1–3.
2. Dawson, C.J.. Interdigital unveils m2m platform compliant with all standards. In: *Machine to Machine Solutions Community eNewsletter*. 2013, .
3. ETSI, . *Machine-to-Machine communications (M2M) ; Functional architecture*. ETSI TS 102 690 V1.1.1 (2011-10); 2011.
4. Guangyu, C., Jian, C., Haisheng, G., ZhiPeng, G., Xuesong, Q.. Oma-dm based mobile device diagnostics and monitoring mechanism. In: *Global Mobile Congress 2009*. 2009, p. 1–6. doi:10.1109/GMC.2009.5295823.
5. Arjona, R.. Mobile device provisioning with syncml. In: *MSDN Magazine*. 2009, .
6. OMA, . *OMA Device Management Protocol*. Open Mobile Alliance; OMA-TS-DM Protocol-V1.2-20060602-C; 2006.
7. OMA, . *OMA Device Management Tree and Description Serialization*. Open Mobile Alliance; OMA-TS-DM TNDS-V1.2-20070209-A; 2007.
8. Hillen, B.A.G., Passchier, I., Matthijssen, E., den Hartog, F., Selgert, F.. Remote management of mobile devices with broadband forum's tr-069. In: *Telecommunications Network Strategy and Planning Symposium, 2008. The 13th International*. ????, .
9. ETSI, . *Machine-to-Machine communications (M2M) ; OMA DM compatible Management Objects for ETSI M2M*. ETSI TS 103 092 V1.1.1 (2012-05); 2012.
10. Alaya, M.B., Monteil, T., Drira, K.. Autonomic framework based on semantic models for self-management of ubiquitous systems. In: *UbiComp*. 2012, p. 860–862.
11. Kephart, J., Chess, D.. The vision of autonomic computing. *Computer* 2003;**36**(1):41–50. doi:10.1109/MC.2003.1160055.
12. OMA, . *Firmware Update Management Object*. Open Mobile Alliance; OMA-TS-DM-FUMO-V1.0.2-20090828-A; 2009.