A functional model extension of OMA device management

Ivaylo I. Atanasov, Evelina N. Pencheva

Machine-to-Machine (M2M) communications allow interworking of smart objects such as sensors, actuators and monitors, and development of innovative applications. Remote entity management provides means of managing M2M device life cycles. Open Mobile Alliance (OMA) defines a device model aimed to support management applications. The OMA device model includes communication attributes of the managed device but does not describe device attributes related to its sensor or actuator functions. The paper presents an extension of OMA device model that includes common sensor and actuator data that might be of interest in the management context. New information structures are defined and described in XML format. The usage of the proposed information structures is illustrated by use cases. State models representing the managing entity's view on the state of M2M device are synthesized. The state models allow scheduling device management tasks, configuring connectivity, firmware updating, performance monitoring, installing and updating software, and managing device capabilities.

Разширение на функционалния модел на ОМА за мениджмънт на устройства (Ивайло Атанасов, Евелина Пенчева). *Комуникациите между устройства (M2M) разрешават взаимодействие на интелигентни обекти като сензори, активиращи устройства и монитори, както и разработване на иновативни приложения. Мениджмънтът на отдалечена единица осигурява средства за мениджмънт на жизнения цикъл на M2M устройство. Open Mobile Alliance (OMA) дефинира модел на устройство, предназначен да подпомага приложения за мениджмънт. Моделът на устройство на ОМА включва комуникационни атрибути на управлявания обект, но не описва атрибути на устройството, които са свързани с неговите сензорни или въздействащи функции. Статията представя разширение на ОМА модела на устройство, което включва общи данни за сензори и активиращи устройства, които могат да представляват интерес в контекста на мениджмънта. Дефинирани са нови структури от данни, които са описани в XML формат. Използването на предложените структури от данни е илюстрирано чрез примери. Синтезирани са модели на състоянията, представящи виждането на управляващата единица. Моделите на състоянията са приложими при мениджмънта на конфигурацията и мениджмънта на неизправностите.*

Introduction

Machine-to-Machine (M2M) communications is a standardized term for interworking of smart objects capable to gather information from the environment without human intervention. The evident growth of M2M communications is estimated to be about 5% share of the global mobile internet traffic in 2018 [1]. This type of communications is expected to have a profound impact on our lives [2]. The large scale deployment of IP-connected smart objects enables development of innovative service in areas like healthcare, transportation and energy. M2M technologies that use Internet Protocol (IP) connectivity are often referred to the notion of "Internet of things" [3].

M2M networks contain heterogeneous communicating devices, support different communication pattern types, feature variety of applications deployed by various providers and operate in a highly dynamic environment. Ubiquitous deployment and tremendous growth of M2M devices is a real challenge to remote device management and provisioning [4]. Device management includes M2M device tracking, remote monitoring and updating as required, as well as provisioning management and reporting of M2M devices on a given network. Rapid configuration and efficient use of network resources require “over the air” management [5].

European Telecommunications Standard Institute (ETSI) standards define the M2M functional architecture [6]. In order to support both M2M device management and M2M application development, ETSI published study on semantic support for M2M data. The aim is to study a new service capability devoted to discover, interpret and use the M2M data from different sources, without any kind of prior knowledge of that. This is essential to offer high-level M2M services and to develop open markets for M2M data [7].

Most of the existing solutions for device management [8], [9], [10], [11] use advanced technologies such as TR 069 of the Broadband Forum and Open Mobile Alliance Device Management (OMA DM). Both protocols feature own specifics. TR 069 CWMP (CPE WAN Management Protocol) is a management protocol used in a cable environment and does not support short messaging, hence, forcing the device to create session management requires other mechanisms [12]. CWMP is a protocol for remote management of end-user devices and as a bidirectional SOAP/HTTP based protocol. It provides the communication between customer premises equipment and auto configuration servers. It is also used as a protocol for remote management of home network devices and terminals, and there is a growing trend for the use in M2M communications. OMA DM is a protocol for management of mobile devices such as mobile phones and tablets that provide functions for software configuration and update, fault and performance management [13], [14]. OMA DM provides a framework that enables device customization and service configuration in a remote fashion. The DM protocol supports operations that allow retrieval of data from a device by means of information presenting a logical Management Object (MO). MO is an XML based structure that accepts commands which in turn trigger certain behaviours in the device. The logical representation of the managed device as a MO allows scheduling and automating of device management tasks, configuring connectivity, updating firmware, diagnosing problems, monitoring performance, installing and updating software, and managing device capabilities. The protocol requires session establishment for device management by sending a special short message to the device.

At the end of 2013, OMA published standards for Lightweight M2M management aiming to support both device management and service logic. The motivation for OMA Lightweight M2M (LWM2M) protocol is to be independent of M2M service logic and to support full device management providing extendability to satisfy specific service logic requirements [15], [16]. The LWM2M technical specification for device object provides a range of device related information which can be queried, and functions for device restart and factory reset. This managed object does not support any sensor or actuator data. Sensors provide information about the physical environment they monitor. Information in this context ranges from the environment property to measurements of the physical state. Sensors can be attached or embedded in the M2M device [17]. Actuators can modify the physical state of the environment like changing the state (translate, rotate, stir, switch on/off etc.) of simple physical entities or activation/deactivation functions of more complex ones [18], [19]. The relation between actuator input and output is usually referred as transfer function. It is responsible for moving or controlling a mechanism or system. The source of energy is typically electric current, hydraulic fluid pressure, or pneumatic pressure. An actuator is the mechanism by which a control system acts upon an environment. The control system can be simple (a fixed mechanical or electronic system) or software-based (e.g. a printer driver, robot control system), as shown in Fig.1.

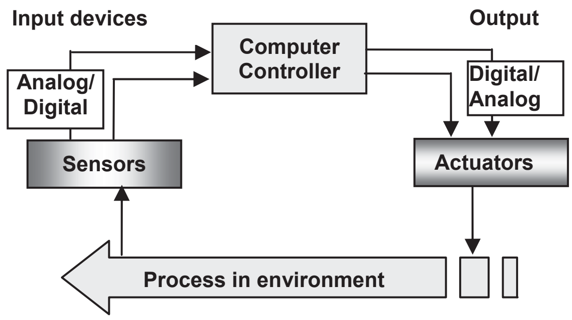

Fig.1. A model of typical control system with sensors and actuators.

This paper describes a range of sensor and actuator related information and suggests XML descriptions which may be defined as a part of M2M device managed object. M2M device state models representing the point of view of the remote entity management are synthesized. Models may be used in configuration management and fault management.

The paper is structured as follows. First, an OMA device managed object is presented. Next, an abstraction of sensor and actuator related information is synthesized and presented by XML definitions. After presenting use cases that illustrate the applicability of the defined data, configuration and fault management state models are described. The conclusion summarizes the contributions.

OMA M2M device management object

OMA DM has a concept of managed objects (MO), which are available on devices and are managed from the server.

The Device LWM2M Object provides a range of device related information, and a device reboot and factory reset function [14]. Each *Device* object is uniquely identified by its *ID*. The Device object is described by the following data. The *Model number* and the *Serial number* are identifiers provided by the manufacturer. The *Firmware version* describes the current firmware version. The *Reboot* item allows the reboot the LWM2M Device to restore the *Device* from unexpected firmware failure. The *Factory reset* enables to perform factory reset of the LWM2M Device to make the LWM2M Device have the same configuration as at the initial deployment. When this Resource is executed, “De-register” operation may be sent to the LWM2M Server(s) before factory reset of the LWM2M Device. The *Available Power Sources* item is described as enumerated type of the following: DC power, internal battery, external battery, power over Ethernet, USB, AC (Mains) power, and solar. The *Power Source Voltage* presents voltage for each *Available Power Sources* instance. The *Power Source Current* presents current for each *Available Power Source*. The *Battery Level* contains the current battery level as a percentage (with a range from 0 to 100). This value is only valid when the value of *Available Power Sources* is internal battery. The *Memory Free* represents the estimated current available amount of storage space which can store data and software in the LWM2M Device (expressed in kilobytes). The *Error Code* presents the current device state. It is described as one of the following: 0=No error; 1=Low battery power; 2=External power supply off; 3=GPS module failure; 4=Low received signal strength; 5=Out of memory; 6=SMS failure; 7=IP connectivity failure; 8=Peripheral malfunction. When the single *Device* Object Instance is initiated, there is only one error code whose value is equal to 0 that means no error. When the first error happens, the LWM2M client changes error code to any non-zero value to indicate the error type. When any other error happens, a new error code instance is created. The reset *Reset Error Code* may be used to delete all error code instances and to create only one zero-value error code that implies no error. The *Current Time* represents the time in the LWM2M client. The *UTC offset* indicates the UTC offset currently in effect for this LWM2M Device. The *Timezone* indicates in which time zone the LWM2M Device is located, in IANA Timezone (TZ) database format. The *Supported Binding and Modes* indicates which bindings and modes are supported in the LWM2M Client.

The description of the OMA device managed object is structured as shown in Fig.2.

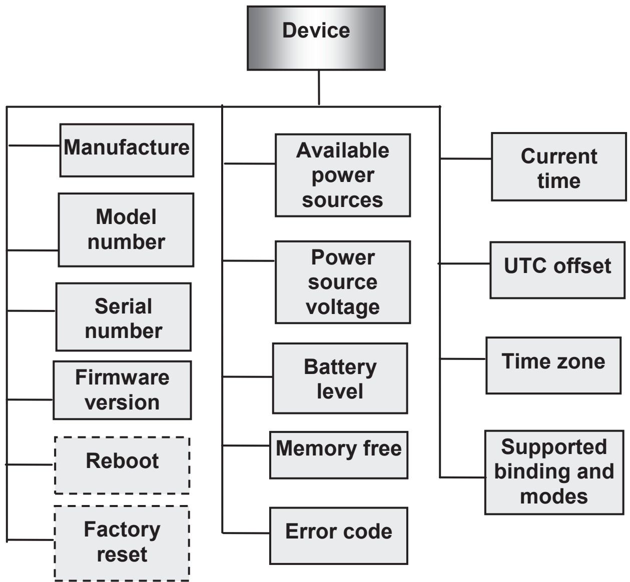

Fig.2. Structure of OMA device managed object.

For each item used to describe the *Device* managed object, operations are defined. The operations indicate what the LWM2M sever is allowed to do with this item. The description of *Device* management object does not contain any information about sensors or actuators related to the device, as well as no M2M application information that may enable application life cycle management.

An extension of OMA M2M device management object

While M2M devices are equipped with communication, memory and elaboration capabilities, the sensing function is essential for their behavior when gather information from the environment. Sensors are used to “feel” the environment by transforming different physical values into their electrical counterpart; actuators do the reverse, they convey an electrical impulse sent by the user to enable or disable a function (open a door, switch a valve, etc.). There are many types of sensors and actuators, and almost as many different standards. Basically, they are divided in two main types: analog and digital. The analog sensors can deliver information as voltage, current, pulse counters, etc. Digital sensors typically include a micro-controller and additional electronics (e.g. analog to digital converters) that perform the conversion of the analog values to a digital output, typically a serial bus. They are also called intelligent sensors.

More detailed information about Device managed object may be obtained by including in its definition the following data.

A M2M device may be equipped with one or more sensors. Fig.3 shows the XML definition for *Sensor Type* item. Each sensor is identified by its ID. There exist different sensor types (e.g. accelerometers, magnetometers, gyroscopes, pressure sensors, humidity sensors, temperature sensors, ambient light sensors, proximity sensors etc.).

```
<Item ID="17">
      <Name>SensorType</Name>
      <Operations>R</Operations>
      <MultipleInstances>Multiple</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type>Integer</Type>
      <RangeEnumeration>0-20</RangeEnumeration>
      <Units></Units>
     <Description><![CDATA[0=accelerometer;
1=magnetometer; 2=gyroscope; 3=pressure sensor;
4=humidity sensor; 5=temperature sensor; 6=ambient light
sensor; 8=proximity sensor
9-20 reserved for other sensor types.]]>
     </Description>
</Item>
```

Fig. 3. XML definition for Sensor Type item.

Common sensor attributes are defined in [17].

The sensitivity of the sensor is defined as the slope of the output characteristic curve or, more generally, the minimum input of physical parameter that will create a detectable output change. In some sensors, the sensitivity is defined as the input parameter change, required to produce a standardized output change. In others, it is defined as an output voltage change for a given change in input parameter.

The sensitivity error is a departure from the ideal slope of the characteristic curve. Fig.4 shows the XML definition for *Sensitivity error* item.

Some sensors perform auto-calibration to correct the accumulated errors at the sensor output. Fig.5 shows the XML definition of the *Autocalibrate* item that may be included in the *Device* object definition.

```
<Item ID="19">
      <Name>Sensitivity Error</Name>
      <Operations>R</Operations>
      <MultipleInstances>Single</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type>Float</Type>
      <RangeEnumeration></RangeEnumeration>
      <Units> </Units>
     <Description><![CDATA[Departure from the ideal slope
of the characteristics curve]]></Description>
</Item>
```

Fig 4. XML definition for Sensitivity Error item.

```
<Item ID="20">
      <Name>Autocalibrate</Name>
      <Operations>E</Operations>
      <MultipleInstances>Single</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type></Type>
      <RangeEnumeration></RangeEnumeration>
      <Units></Units>
      <Description><![CDATA[Execute to restore the Device's
accuracy.]]>
      </Description>
</Item>
```

Fig. 5. XML definition for Autiocalibrate item.

The concept of precision refers to the degree of reproducibility of a measurement. In other words, if exactly the same value were measured a number of times, an ideal sensor would output exactly the same value every time. But real sensors output a range of values distributed in some manner relative to the actual correct value.

The accuracy of the sensor is the maximum difference that will exist between the actual value (which must be measured by a primary or good secondary standard) and the indicated value at the output of the sensor. The accuracy can be expressed either as a percentage of full scale or in absolute terms. Some sensors allow dynamic accuracy set. Fig.6 shows the XML definition of the *Current Accuracy* item.

The range of the sensor is the maximum and minimum values of applied parameter that can be measured. The dynamic range is the total range of the sensor from minimum to maximum.

The resolution is the smallest detectable incremental change of input parameter that can be detected in the output signal. Resolution can be expressed either as a proportion of the reading (or the full-scale reading) or in absolute terms.


```
<Item ID="21">
      <Name>Current Accuracy</Name>
      <Operations>RW</Operations>
      <MultipleInstances>Single</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type>Integer</Type>
      <RangeEnumeration>0-100</RangeEnumeration>
      <Units>%</Units>
      <Description><![CDATA[Current device
accuracy]]></Description>
</Item>
```


Fig. 6. XML definition for Current Accuracy item.

The offset error of a transducer is defined as the output that will exist when it should be zero or, alternatively, the difference between the actual output value and the specified output value under some particular set of conditions.

Sensors do not change output state immediately when an input parameter change occurs. Rather, it will change to the new state over a period of time, called the response time. The response time can be defined as the time required for a sensor output to change from its previous state to a final settled value within a tolerance band of the correct new value. This concept is somewhat different from the notion of the time constant (T) of the system. This term can be defined in a manner similar to that for a capacitor charging through a resistance and is usually less than the response time.

A M2M device may be equipped with one or more actuators. Each actuator is identified by its identifier. Fig.7 shows an XML definition for *ActuatorType* item.


```
-<Item ID="17">
      <Name>ActuatorType</Name>
      <Operations>R</Operations>
      <MultipleInstances>Single</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type>Integer</Type>
      <RangeEnumeration>0-3</RangeEnumeration>
      <Units></Units>
     <Description><![CDATA[0=hydraulic; 1=pneumatic;
2=electric; 3=mechanical.]]></Description>
</Item>
```


Fig.7. XML definition for Actuator Type item

Actuators are characterized by their resolution. Resolution is number of possible output levels the digital to analogue converter is designed to reproduce and it is usually stated as the number of bits it uses, which is the base two logarithm of the number of levels. Actuator sensitivity is defined as the change in output for a given change in input, usually a unit change in input. Sensitivity represents the slope of the transfer function and it is measured in units of output quantity per units of input quantity.

Actuators are also described by their performance metrics that include speed, acceleration, and force. As far as the current actuator force may be changed within a preliminary defined range, it may be of interest from the management point of view.

Fig.8 shows an XML definition for the *Maximum Actuator Static Force* item, and Fig.9 shows an XML definition for the *Current Actuator Static Force* item. The managing entity is allowed only to read the value of actuator maximum force, while the actuator current force may be changed also.


```
<Item ID="18">
      <Name>ActuatorMaxForce</Name>
      <Operations>R</Operations>
      <MultipleInstances>Single</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type>Float</Type>
      <RangeEnumeration></RangeEnumeration>
      <Units>N</Units>
     <Description><![CDATA[describes maximum actuator
force]]></Description>
</Item>
```


Fig.8. XML definition for Actuator Max Force item.


```
<Item ID="19">
      <Name>ActuatorCurrentForce</Name>
      <Operations>RU</Operations>
      <MultipleInstances>Single</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type>Float</Type>
      <RangeEnumeration></RangeEnumeration>
      <Units>N</Units>
     <Description><![CDATA[describes current actuator
force]]></Description>
</Item>
```


Fig.9. XML definition for Actuator Current Force item

Just like the sensors, the actuators might be capable of performing auto-calibration in order to cope with any possible accumulated errors.

The mode of operation determines how the M2M device equipped with sensors reports its measurements. An M2M device may be configured to send measurements periodically or triggered (on occurrence of specific events) and it is also possible to request reporting on demand. Fig.10 shows the XML definition for the *Operation mode* item.

Use cases

The OMA Device Management and Service Enable Interface are used by the LWM2M Server to access Object Instances and Resources available from the LWM2M Client. The interface provides this access through the use of “Create”, “Read”, “Write”, “Delete”, “Execute”, “Write Attributes”, or “Discover” operations. The operations that resource supports are defined in the Object definition.


```
<Item ID="20">
      <Name>OperationMode</Name>
      <Operations>RW</Operations>
      <MultipleInstances>Multiple</MultipleInstances>
      <Mandatory>Optional</Mandatory>
      <Type>Integer</Type>
      <RangeEnumeration>0-10</RangeEnumeration>
      <Units></Units>
     <Description><![CDATA[0=idle;  1=periodic;  2=triggered;
3=periodic and triggered; 4-10 reserved for other operation
modes.]]>
     </Description>
</Item>
```


Fig.10. XML definition for Operation Mode item

The “Read” operation is used to access the value of a Resource, an array of Resource Instances, an Object Instance or all the Object Instances of an Object. The “Write” operation is used to change the value of a Resource, an array of Resources Instances or multiple Resources from an Object Instance. Fig.11 shows how the LWM2M Server changes the *Sensor Accuracy* of the LWM2M Client. The “Execute” operation is used by the LWM2M Server to initiate some action, and can only be performed on individual resources. Fig.12 shows how the LWM2M Server activates auto-calibration on the LWM2M Client.

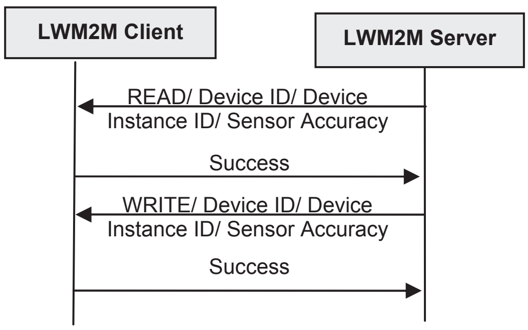

Fig. 11. LWM2M Server changes the Sensor Accuracy attribute of the Device object

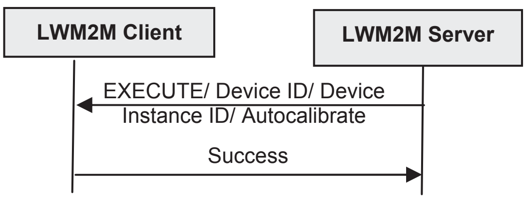

Fig. 12. LWM2M Server requests execution of sensor auto-calibration by the Device object.

State machines for configuration and fault management

REST (representational state transfer) is an architectural style that relies on HTTP. It supports the concept that every physical and/or logical entity is a *resource* that has a particular *state* that can be manipulated [20]. This concept maps naturally to the M2M devices, which can be read or configured. REST-based architecture features several advantages that may be useful for programmability of sensors and actuators by virtual thing and manipulating of parameters through web browser. In this section, state models related to M2M device configuration and fault management are presented.

The M2M device state machine from configuration management point of view is shown in Fig.13.

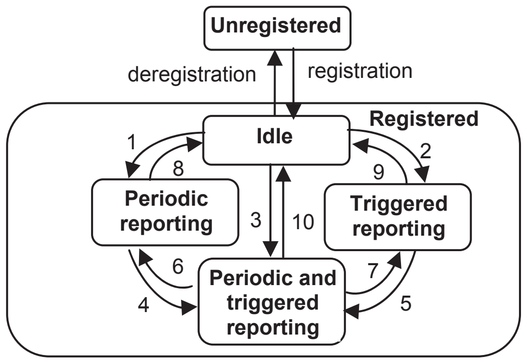

Fig. 13. Configuration management state model

In *Unregistered* state, the device is switched off and does not provide any service. When the M2M device is switched on it performs registration, which bounds its current IP address with its identification, and moves to *Idle* state. In *Idle* state, the LWM2M Server may configure the M2M device to operation in different operation modes. In *Periodic Reporting* state, the M2M device is configured to send reports periodically. In *Triggered Reporting* state, the M2M device is configured to send reports when a particular event occurs. In *Periodic and Triggered Reporting* state, the M2M device is configured to report measurements both periodically and event driven. State transitions occur when the LWM2M Server writes a new value of the *Operation Mode* resource attribute. For example, the transition 1 from *Idle* state to *Periodic Reporting* state occurs when the LWM2M Server writes the value of 1 of the *Operation Mode* attribute, and the transition 8 occurs when the *Operation Mode* attribute value is set to 0.

The M2M device state machine from fault management point of view is shown in Fig.14. In *Operational* state, the device is in operation and functions normally. When the *Error Code* value is different from 0, or due to administrative reasons, the LWM2M Server may invoke the Block operation on the M2M device and a transition to *Blocked* state occurs.

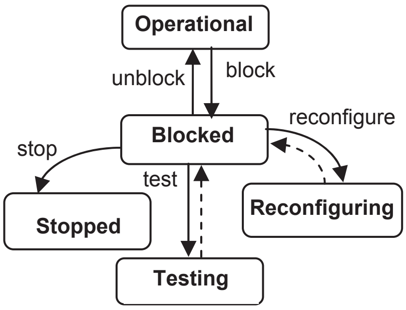

Fig. 14. Fault management state model

In *Blocked* state, the M2M device is out of operation. Some error may be removed by device reconfiguration or restart. In *Reconfiguring* state, the device is under reconfiguration. In *Testing* state, the M2M device is under testing. The aim of the test is to check the M2M device performance. If the fault is temporary and the test results shows that the error is removed, the LWM2M Server may invoke the Unblock operation in order to restore the operational state of the device. If the fault is stable and can not be removed by remote operations, the LWM2M Server invokes the Stop operation and the M2M device is stopped. The test results become available to the LWM2M Server by reading the *Error Code* value. The Unblock, Block, Reconfigure, Stop and Test operations may be defined as M2M device attributes in a similar way as those of e.g. the *Autocalibrate* item.

Conclusion

The logical representation of a resource under management is by a managed object. OMA Lightweight M2M definition of Device managed object includes a range of device related information which can be queried by the LWM2M Server, and a device reboot and factory reset function. The device is described by number of items including failure status but no sensor, actuator or application attributes are presented. As far as the OMA standards are open and allow definitions of other management objects and extension of the existing ones, the paper defines an extension of information used to describe a M2M object defined in the context of M2M device management. The extension includes sensor and actuators attributes that may be of interest to the managing entity. The sensor and actuator attributes are described by XML definitions following the OMA object definition pattern.

The paper presents also synthesized state models reflecting the configuration and fault management point of view of the managing entity. The state models allow scheduling device management tasks, configuring connectivity, firmware updating, performance monitoring, installing and updating software, and managing device capabilities.

Acknowledgements

The work is supported by the research project №141ПР004-7, funded by RDS, TU-Sofia.

REFERENCES

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013-2018, Retrieved from: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html

[2] Potsch, T., S. Marwat, Y. Zaki, C. Gorg. Influence of future M2M communication on the LTE system. Proc. of Wireless and Mobile Networking Conference (WMNC), 2013, pp.1-4.

[3] H. Zhou. The Internet of Things in the Cloud, A Middleware Perspective, CRC Press, 2012.

[4] Kostadinova, S., R. Dimova. Performance Parameters Evaluation in Broadband Access Network, Proceedings of 14th International Conference on Optimization of Electrical and Electronic Equipment, OPTIM'2014, 2014, Brasov, Romania, pp. 790-795.

[5] Vasseur, J.-P, A. Dunkels. Interconnecting Smart Objects with IP, The next Internet. Morgan Kaufmann, Elsevier, 2010.

[6] ETSI TS 102 690 Machine-to-Machine (M2M) Functional architecture, v.1.1.1, 2010.

[7] ETSI TR 101 584 Machine-to-Machine Communications (M2M); Study on Semantic support for M2M Data, v2.1.1, 2013.

[8] Chang, Y. C., T.Y. Chi, W.C. Wang, S.Y. Kuo Dynamic software update model for remote entity management of machine-to-machine service capability. IET Communications, vol.7, issue 1, 2013, pp.32-39.

[9] Abdalla, I., S. Venkatesan. Remote subscription management of M2M terminals in 4G cellular wireless networks, Proceedings of IEEE International Conference on Local Computer Networks Workshop, 2012, Florida, USA, pp. 877-885.

[10] Cackovic, V., Z. Popovic. Abstraction and Semantics support in M2M communications, Proceedings of International Convention on Information &

Communication Technology & Microelectronics MIPRO, 2013, pp.404-408.

[11] L. Barriga, B. Smeets, K. Sallberg. M2M Remote-Subscription Management, Ericsson Review, 2011.

[12] Broadband Forum, TR-069 CPE WAN Management Protocol, version 1.4, Technical report, 2013.

[13] M. Unmehopa. Management and Provisioning of M2M Devices for a Connected World. ETSI M2M Workshop, 2011.

[14] OMA-AD-DM-V1_3 Device Management Architecture, version 1.3, 2010.

[15] OMA LightweightM2M v1.0, 2013, Retrieved from: http://technical.openmobilealliance.org/Technical/release_program /lightweightM2M_v1_0.aspx

[16] OMA-SUP-XML_LWM2M_Device-V1_0-20131210-C, Retrieved from: http://technical.openmobilealliance.org/Technical/release_program/docs/LightweightM2M/V1_0-20131210-C/OMA-SUP-XML_LWM2M_Device-V1_0-20131210-C.txt

[17] Carr, J., J. Brown. Introduction to Biomedical Equipment Technology, Third Edition, prentice Hall, 2010.

[18] Yaguchi, H., Y. Nanjo, K. Ishikawa. Considerations of the performance characteristics of the cables micro-actuator by using mechanical DC-AC inverter, Proceedings of European Conference on Power electronics and Applications, 2007, pp. 1-5.

[19] Rakotondrabe, M. Smart Materials-based Actuators at the Micro/Nano-Scale, Characterization, Control, and Applications. Springer, 2013.

[20] Rouached, M, H. Sallay. RESTful Web Services for High Speed Intrusion Detection Systems, Proceedings of IEEE International Conference on Web Services ICWS'2013, pp.621-622.

Prof. Dr. Ivaylo I. Atanasov *is with Faculty of Telecommunications, Technical University of Sofia. His research interests are in service development platforms and wireless applications.*

tel.:+359 2 965 2050 *e-mail: iia@tu-sofia.bg*

Prof. D.Sc. Evelina N. Pencheva *is with Faculty of Telecommunications, Technical University of Sofia. Her research interests are in service development platforms and multimedia networks.*

tel.+359 2 965 3695 *e-mail: enp@tu-sofia.bg*

Received on: 30.05.2014