Models of M2M Device Management

IVAYLO ATANASOV, EVELINA PENCHEVA
Faculty of Telecommunications
Technical University of Sofia
8 Kliment Ohridski blvd, Sofia
BULGARIA
iia@tu-sofia.bg; enp@tu-sofia.bg

Abstract: - Machine-to-Machine (M2M) stands for networking of machines and devices that gather information from their environment and share it over the communication network. Devices must be set up and configured correctly, and they need to use available network bearers efficiently. The growth of connected devices makes the device management a challenging task. Reduction in M2M device deployment time and operational costs may be achieved by automation of management processes. In this paper, we propose context-ware models for connectivity management and study aspects of autonomous behaviour in the context of bearer selection procedure based on policies. Connectivity management models are formally described and verified using the concept of weak bi-simulation. The autonomous behaviour which includes monitoring of device connectivity parameters and policy-based bearer selection is modelled and formalized by temporal logic. The validation process is based on a suit of unit tests that allow comparing the expected message exchange traces to the observed ones.

Key-Words: - Machine-to-Machine communications, Connectivity management, Finite state machines, Formal verification, Weak bi-simulation, Autonomous agent, REST

1 Introduction

Machine-to-Machine (M2M) communications have various application areas in almost any environment. Despite the differences, all these areas set common requirements for connected devices. Devices must be set up and configured correctly, and they need to use available network bearers efficiently [1]. The increased amount of equipment and the explosion of M2M services become business and technical challenges for network operators [2].

Device management includes functions like automated device configuration, over-the-air firmware updates, remote reboots, remote diagnostics and troubleshooting, security and integrity. Different protocols and proprietary solutions have fragmented the M2M market and have added complexity, time and cost to integration process [3]. The variety of platforms addressing different activation, billing, monitoring and device management functions do not provide an abstraction required for scalable platform that adheres to standards and addresses a broad range of common M2M functions [4]. Such an abstraction is provided by OMA Lightweight M2M [5].

Lightweight M2M (LWM2M) is a protocol from the Open Mobile Alliance (OMA) for M2M device management. It defines device management procedures between a LWM2M server in a cloud and a LWM2M client, which is located in a device.

In this paper, we study aspects of device connectivity management. The motivation for the research is that the device connectivity management comprises complex operations that are quite different from the application business logic. It is a complex task due to a large and growing category of connected devices with limited computing power and memory, and limited battery lifetime. Devices may be connected using cellular bearers such as GSM, TD-SCDMA, WCDMA, CDMA2000, WiMAX, or LTE, wireless bearers like WLAN, Bluetooth or IEEE 802.15.4, or may use wire line ones as Ethernet, DSL or PLC. Monitored connectivity parameters include the line voltage and signal strength at the device side [6]. Different technologies have different requirements for quality of service (QoS), which complicates the logic for bearer selection. Furthermore, the logic for bearer selection may be based on different policies such as the device location and the account balance of the M2M device provider in case of prepaid payment.

The reduction of device connectivity management complexity can be achieved by embedding autonomic features in operation support systems [7]. The autonomic system exposes reactive or proactive behaviour based on external stimuli, following goals that are required to fulfil, policies, capabilities, principles of operation, experience and knowledge.

To mitigate the issues related to device management, we propose a connectivity management model which is compliant with OMA Lightweight M2M. The model reflects both the device and server views on connectivity management. It includes details related to configuration of observation procedure, notification about monitored parameters, bearer change, and configuration of a new Access Point Name (APN). In addition, we propose a model of autonomous agent responsible for device connectivity management. The agent observes device connectivity parameters and based on preliminary defined policies determines the best bearer that has to be used by the device. Models are formally described and verified. The model validation is based on the Google's REST toolkit.

The paper is organized as follows. In the next section, we discuss in brief the related work. Section 3 presents the client and server views on device connectivity management and a method for formal verification of the models. Section 4 studies autonomic feature of device connectivity management and describes the knowledge base of an autonomous agent that controls bearer selection for M2M devices. In Section 5, the validation process based on RESTful architecture is discussed. The conclusion summarizes the author's contribution.

2 Related Work

In [8], the authors investigate how existing IP-based network management protocols can be implemented on resource-constrained devices. A lightweight RESTful Web service approach to enable device management of wireless sensor devices, based on constrained application protocol is proposed in [9]. In [10], the authors use field device integration technology to achieve seamless maintenance by cooperation of device management systems and computerized maintenance management systems. In [11], the authors present an out-of-box device management to automatically add and remove devices from the system, based on the connectivity. The Ericsson Device Connection Platform which provides the operator with access to key functionalities to manage the connectivity of the M2M business, including device management, subscription management and self-service is discussed in [12]. In [13], the authors describe a smart M2M gateway based architecture to manage the huge volume of M2M devices and endpoints, which is compliant with both ETSI and one M2M standards recommendations. A solution for dynamical provisioning of communication parameters between M2M endpoints using a device management protocol is presented in [14], where OMA LWM2M was chosen for its energy efficiency. An M2M service platform architecture of a home automation system is proposed in [15] which M2M service enablement, M2M device management, and M2M communication management subsystems. In [16], the authors propose a dynamic device connection method that can connect services with devices located close to users by installing the device drivers and/or protocol adapters dynamically.

The proposed solutions, based on LWM2M, consider high level architectural aspects and do not provide details on behavioural models that follow the M2M device management procedures. In this paper, we suggest an approach to formal verification of LWM2M server and client behaviour related to device management.

Currently, there is a lot of work conducted on autonomics by the research community. In [17], the authors discuss challenges and enablers that allow connected machines to evolve and act in a more autonomous way and propose architectural approach based on situational knowledge acquisition and analysis techniques in order to make machines aware of conditions and events affecting systems behaviour. In [18], the authors propose a middleware architecture that connects the appropriate devices and applications, and is based on software agents representing devices and applications negotiating between each other on the terms by which the data can be used. In [19], the authors propose network architecture for remote monitoring and surveillance M2M networks with broadband satellite connection. In [20], it is proposed a flexible multi-agent approach, leveraging semantic-based resource discovery and orchestration for home and building automation applications. In [21], a generic architecture for multi-goal, adaptable and open autonomic systems, exemplified via the development of a concrete autonomic application for the smart micro-grid is proposed. Cognitive and mathematical models of data, information, knowledge, and intelligence are proposed in [22]. In [23], the authors present methodology for formal verification of hardware security requirements of remote attestation architecture for embedded systems. In [24], the authors claim that agent-based, adaptive Parallel and

Distributed Simulation (PADS) approaches are needed, together with multi-level simulation of machine type communications, which provide means to perform highly detailed simulations, on demand. A dynamic service arbitration scheme based on autonomic computing, which allows only selected devices to be utilized instead of all deployed devices, is proposed in [25].

While presenting high level architectural aspects of autonomous systems, these works discuss proprietary solutions and do not consider autonomics in generic M2M communications.

3 M2M Device Connectivity Management

3.1 Connectivity management models as seen by the server and device

Typical sequence of procedures performed by the server and device in the context of connectivity management is as follows.

1. The server establishes an observation relationship with the device to acquire periodical or triggered notifications about line voltage and signal strength.
2. The device sends periodical or triggered notifications about line voltage and signal strength.
3. The server queries about used and available network bearers.
4. The server initiates bearer selection.
5. The server queries about connectivity parameters.
6. The server creates and enables a new APN profile.
7. The server cancels observation.

Deployment of LWM2M requires modeling of state machines maintained in the device and in the network. In the following sections we model the behavior in the context of M2M connectivity management and using formal models' description we provide functional verification of the proposed models. The aim is to prove that the M2M device (client) and the remote server are synchronized.

The connectivity management model as seen by the server is shown in Fig.1.

In Operational$_S$ state, the device is registered and operational. In ObservationConfiguration state, the server sets observation policy in the device. In ObservationAck state, the server waits for acknowledgment that the observation is active. In QueryNetConnectivity state, the server has sent a query for device connectivity parameters and waits for the requested information. In PreferredBearerAck state, the server has requested preferred bearer selection and waits for the acknowledgement. In BearerReregistration state, the server waits for device re-registration after bearer selection. In APNprofile state, the server has requested creation of new APN profile and waits for acknowledgement. In APNReregistration state, the server waits for device re-registration after new APN profile selection. In APNactivation state, the server has activated the new APN and waits for acknowledgement. In QueryAPNConnectivity state, the server waits for the requested information about APN connectivity. In CancelAck state, the server has cancelled the observational relationship and waits for acknowledgement.

We use the notation of Labeled Transition System (LTS) to formally describe the model.

By $CM_S = (S_S, Act_S, \rightarrow_S, s_0^S)$ it is denoted an LTS representing the server's view on connectivity management state model as follows:

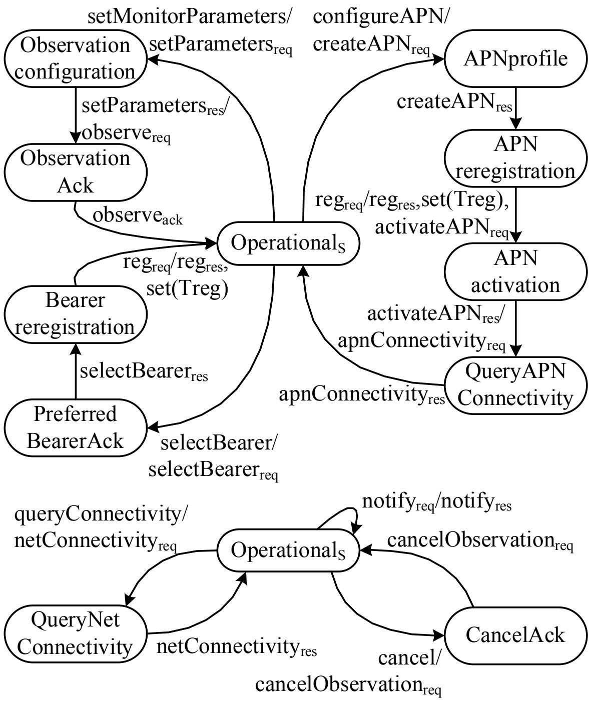


Fig.1 Connectivity management model as seen by the server

S_S ={Operational$_S$, ObservationConfiguration, ObservationAck, QueryNetConnectivity, PreferredBearerAck, BearerReregistration, APNReregistration, APNprofile, APNactivation, QueryAPNConnectivity, CancelAck};

Act_{S} = {queryConnectivity, netConnectivity$_{\mathrm{res}}$, setMonitorParameters, notify$_{\mathrm{req}}$, setParameters$_{\mathrm{res}}$, observe$_{\mathrm{ack}}$, selectBearer, selectBearer$_{\mathrm{res}}$, reg$_{\mathrm{req}}$, configureAPN, createAPN$_{\mathrm{res}}$, activateAPN$_{\mathrm{res}}$, apnConnectivity$_{\mathrm{res}}$, cancel, cancelObservation$_{\mathrm{res}}$};

$\rightarrow_{\mathrm{S}} = \{ \tau_1^S, \tau_2^S, \tau_3^S, \tau_4^S, \tau_5^S, \tau_6^S, \tau_7^S, \tau_8^S, \tau_9^S, \tau_{10}^S, \tau_{11}^S, \tau_{12}^S, \tau_{13}^S, \tau_{14}^S, \tau_{15}^S, \tau_{16}^S \}$;

s_0^{S} = { Operational$_{\mathrm{S}}$ }.

where

τ_1^S = (Operational$_{\mathrm{S}}$ setMonitorParameters ObservationConfiguration),

τ_2^S = (ObservationConfiguration setParameters$_{\mathrm{res}}$ ObserveAck),

τ_3^S = (ObserveAck observe$_{\mathrm{ack}}$ Operational$_{\mathrm{S}}$),

τ_4^S = (Operational$_{\mathrm{S}}$ notify$_{\mathrm{req}}$ Operational$_{\mathrm{S}}$),

τ_5^S = (Operational$_{\mathrm{S}}$ queryConnectivity QueryNetConnectivity),

τ_6^S = (QueryNetConnectivity netConnectivity$_{\mathrm{res}}$ Operational$_{\mathrm{S}}$),

τ_7^S = (Operational$_{\mathrm{S}}$ selectBearer PreferredBearerAck),

τ_8^S = (PreferredBearerAck selectBearer$_{\mathrm{res}}$ BearerReregistration),

τ_9^S = (BearerReregistration reg$_{\mathrm{req}}$ Operational$_{\mathrm{S}}$),

τ_{10}^S = (Operational$_{\mathrm{S}}$ configureAPN APNprofile),

τ_{11}^S = (APNprofile createAPN$_{\mathrm{res}}$ APNReregistration),

τ_{12}^S = (APNReregistration reg$_{\mathrm{req}}$ APNactivation),

τ_{13}^S = (APNactivation activateAPN$_{\mathrm{res}}$ QueryAPNConnectivity),

τ_{14}^S = (APNactivation apnConnectivity$_{\mathrm{res}}$ Operational$_{\mathrm{S}}$),

τ_{15}^S = (Operational$_{\mathrm{S}}$ cancel CancelAck),

τ_{16}^S = (CancelAck cancelObservation$_{\mathrm{res}}$ Operational$_{\mathrm{S}}$).

The connectivity management model as seen by the device is shown in Fig.2.

In Operational$_{\mathrm{D}}$ state, the device is registered and operational. In this state, the server may set the observation policy and activate observation, as well as it may cancel observation. In Operational$_{\mathrm{D}}$ state, the device sends responses of queries on network connectivity. In NotifyAck state, the device has notified the server about requested information and waits for response. In UpdateBearer state, the device is in a process of used network bearer switching and re-registration. In APNconfiguration state, the device is in a process of creation and enablement of a new APN profile.

By CM_{D}= (S_{D}, Act_{D}, $\rightarrow_{\mathrm{D}}$, s_0^{D}) it is denoted an LTS representing the device's view on connectivity management state model as follows:

S_D = { Operational$_{\mathrm{D}}$, NotifyAck, UpdateBearer, APNconfiguration };

Act_D = { setParamaters$_{\mathrm{req}}$, observe$_{\mathrm{req}}$, netConnectivity$_{\mathrm{req}}$, cancelObservation$_{\mathrm{req}}$, T$_{\mathrm{mon}}$, trigger, notify$_{\mathrm{res}}$, selectBearer$_{\mathrm{req}}$, reg$_{\mathrm{res}}$, activateAPN$_{\mathrm{req}}$, create$_{\mathrm{req}}$, apnConnectivity$_{\mathrm{res}}$ };

$\rightarrow_D = \{ \tau_1^D, \tau_2^D, \tau_3^D, \tau_4^D, \tau_5^D, \tau_6^D, \tau_7^D, \tau_8^D, \tau_9^D, \tau_{10}^D, \tau_{11}^D, \tau_{12}^D, \tau_{13}^D \}$

s_0^D = { Operational$_{\mathrm{D}}$ },

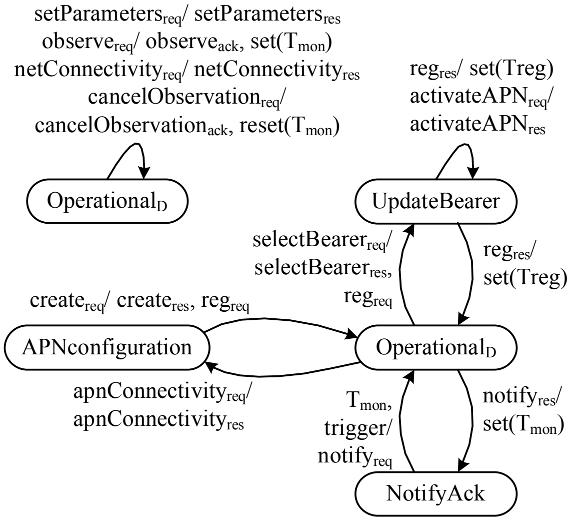

setParameters$_{\mathrm{req}}$/ setParameters$_{\mathrm{res}}$
observe$_{\mathrm{req}}$/ observe$_{\mathrm{ack}}$, set(T$_{\mathrm{mon}}$)
netConnectivity$_{\mathrm{req}}$/ netConnectivity$_{\mathrm{res}}$
cancelObservation$_{\mathrm{req}}$/
cancelObservation$_{\mathrm{ack}}$, reset(T$_{\mathrm{mon}}$)
reg$_{\mathrm{res}}$/ set(Treg)
activateAPN$_{\mathrm{req}}$/
activateAPN$_{\mathrm{res}}$
Operational$_{\mathrm{D}}$
UpdateBearer
selectBearer$_{\mathrm{req}}$/
selectBearer$_{\mathrm{res}}$,
reg$_{\mathrm{req}}$
reg$_{\mathrm{res}}$/
set(Treg)
create$_{\mathrm{req}}$/ create$_{\mathrm{res}}$, reg$_{\mathrm{req}}$
APNconfiguration
Operational$_{\mathrm{D}}$
apnConnectivity$_{\mathrm{req}}$/
apnConnectivity$_{\mathrm{res}}$
T$_{\mathrm{mon}}$,
trigger/
notify$_{\mathrm{req}}$
notify$_{\mathrm{res}}$/
set(T$_{\mathrm{mon}}$)
NotifyAck

Fig.2 Connectivity management model as seen by the device

where

τ_1^D = (Operational$_{\mathrm{D}}$ setParamaters$_{\mathrm{req}}$ Operational$_{\mathrm{D}}$),

τ_2^D = (Operational$_{\mathrm{D}}$ observe$_{\mathrm{req}}$ Operational$_{\mathrm{D}}$),

τ_3^D = (Operational$_{\mathrm{D}}$ netConnectivity$_{\mathrm{req}}$ Operational$_{\mathrm{D}}$),

τ_4^D = (Operational$_{\mathrm{D}}$ cancelObservation$_{\mathrm{req}}$ Operational$_{\mathrm{D}}$),

τ_5^D = (Operational$_{\mathrm{D}}$ selectBearer$_{\mathrm{req}}$ UpdateBearer),

τ_6^D = (UpdateBearer reg$_{\mathrm{res}}$ Operational$_{\mathrm{D}}$),

τ_7^D = (Operational$_{\mathrm{D}}$ T$_{\mathrm{mon}}$ NotifyAck),

τ_8^D = (Operational$_{\mathrm{D}}$ trigger NotifyAck),

τ_9^D = (NotifyAck notify$_{\mathrm{res}}$ Operational$_{\mathrm{D}}$),

τ_{10}^{D} = (Operational$_{\mathrm{D}}$ create$_{\mathrm{req}}$ APNconfiguration)

τ_{11}^{D} = (APNconfiguration reg$_{\mathrm{res}}$ APNconfiguration)

τ_{12}^{D} = (APNconfiguration activateAPN$_{\mathrm{req}}$ APNconfiguration)

τ_{13}^{D} =(APNconfiguration apnConnectivity$_{\mathrm{req}}$ Operational$_{\mathrm{D}}$).

In order to prove that both state machines are synchronized, we use the concept of weak bisimulation.

3.2 Formal verification of Connectivity Management models

Intuitively, in terms of observed behaviour, two state machines have bi-similar relation if one state machine displays a final result and the other state machine displays the same result [26]. Strong bi-similarity requires existence of homomorphism between transitions in both state machines. In practice, strong bi-similarity puts strong conditions for equivalence which are not always necessary. For example, internal transitions can present actions, which are internal to the system (i.e. not observable). In weak bi-similarity, internal transitions can be ignored. The concept of weak bi-similarity is used to study the modelling aspects of M2M device registration.

We will use the following notations:

- $s \overset{a}{\rightarrow} s'$ stands for the transition (s, a, s');
- $s \overset{a}{\rightarrow}$ means that ∃ s': $s \overset{a}{\rightarrow} s'$;
- $s \overset{\mu}{\Longrightarrow} s_n$, where $\mu = a_1, a_2, ..., a_n$: ∃ $s_1, s_2, \ldots, s_n$,

such that $s \overset{a_1}{\rightarrow} s_1 \ldots \overset{a_n}{\rightarrow} s_n$;

- $s \overset{\mu}{\Longrightarrow}$ means that ∃ s', such as $s \overset{\mu}{\Longrightarrow} s'$;
- $\overset{\hat{\mu}}{\Rightarrow}$ means $\Rightarrow$ if $\mu \equiv \tau$ or $\overset{\mu}{\Rightarrow}$ otherwise,

where τ is one or more internal (invisible) actions.

Definition 2: Two labelled transition systems $T = (S, A, \rightarrow, s_0)$ and $T' = (S', A, \rightarrow', s_0')$ are *weakly bi-similar* $(T \sim T')$ if there is a binary relation $U \subseteq S \times S'$ such that if $s_1\ U\ t_1 : s_1 \subseteq S$ and $t_1 \subseteq S'$ then $\forall a \in Act$:

- $s_1 \overset{a}{\Rightarrow} s_2$ implies ∃ $t_2 : t_1 \overset{\hat{a}}{\Rightarrow}' t_2$ and s_2 U t_2;
- $t_1 \overset{a}{\Rightarrow}' t_2$ implies ∃ s_2: $s_1 \overset{a}{\Rightarrow} s_2$ and s_2 U t_2.

So, in order to prove that considered LTSs expose equivalent behaviour, it is necessary to identify a bi-similar relation between their states that satisfies the above conditions.

Proposition: The labelled transition systems CM_{S} and CM_{D} are weakly bisimilar.

Proof: To prove that both LTSs bisimulate each other it is necessary to identify a bisimilar relation between their states.

Let U_{DS} = {(Operational$_{\mathrm{D}}$, Operational$_{\mathrm{S}}$),
(UpdateBearer, PreferredBearerAck),
(APNconfiguration, APNprofile)},

then:

1. For Operational$_{\mathrm{D}}$ ∃{ τ_1^D , τ_2^D } and for Operational$_{\mathrm{S}}$ ∃ { τ_1^S , τ_2^S , τ_3^S } - setting observation policy;
2. For Operational$_{\mathrm{D}}$ ∃{ τ_7^D , τ_9^D } and for Operational$_{\mathrm{S}}$ ∃{ τ_4^S } - periodic reporting of signal strength and line voltage;
3. For Operational$_{\mathrm{D}}$ ∃{ τ_8^D , τ_9^D } and for Operational$_{\mathrm{S}}$ ∃ { τ_4^S } – the same as in 2 but triggered case;
4. For Operational$_{\mathrm{D}}$ ∃{ τ_4^D } and for Operational$_{\mathrm{S}}$ ∃ { τ_{15}^S, τ_{16}^S } - observation cancelation;
5. For Operational$_{\mathrm{D}}$ ∃{ τ_3^D } and for Operational$_{\mathrm{S}}$ ∃ { τ_5^S , τ_6^S } - network connectivity queries;
6. For Operational$_{\mathrm{D}}$ ∃{ τ_5^D } and for Operational$_{\mathrm{S}}$ ∃ { τ_7^S } – preferred bearer selection;
7. For UpdateBearer ∃{ τ_6^D } and for PreferredBearerAck ∃{ τ_8^S , τ_9^S } – re-registration after preferred bearer selection;
8. For Operational$_{\mathrm{D}}$ ∃{ τ_{10}^D } and for Operational$_{\mathrm{S}}$ ∃ { τ_{10}^S } – creation a new APN profile;
9. For APNconfiguration ∃{ $\tau_{11}^D, \tau_{12}^D, \tau_{13}^D$ } and for APNprofile ∃ { $\tau_{11}^S, \tau_{12}^S, \tau_{13}^S, \tau_{14}^S$ } – re-registration, APN activation and checking the APN connectivity.

Therefore CM_{D} and CM_{S} are weakly bisimilar, which means that both state machines, representing the server and device views on connectivity management, are synchronized.□.

4 Adding Intelligence to Device Connectivity Management

4.1 OMA Trap Framework

The control logic for device connectivity management is complex because the bearer selection procedure may depend on multiple factors. Connectivity management is a part of diagnostics and monitoring function. A device can be remotely invoked to execute a diagnostics related logic and to return results. For the aims of connectivity management, the remote server may employ a trap mechanism to enable the device to capture and report events and other relevant information related to device connectivity. Each event that is specified as a trap is assigned an identifier. If the device supports a trap, it means that the device is capable of monitoring the event and sending notifications whenever it detects the event.

OMA DiagMon Trap Events specification defines a number of standardized traps [27], [28]. OMA traps that may be used for connectivity management are geographic traps, received power trap, call drop trap, QoS trap, and data speed trap. Geographic trap may be used for location based bearer selection. It goes to active when a device enters into a specific geographic area. Whenever the device leaves that specific geographic area, the trap goes to inactive. The received power trap may be used for bearer selection based on received signal strength at the device. It can helpful in connectivity optimization process when the received power of the device drops below the server-specific value. Whenever a device's received power drops below an agent-specified value (TrapActivePower), it causes this trap to go active. Alternatively, when device senses power rises above another agent-specified value (TrapInactivePower), it causes this trap to go inactive. In cases that the trap goes active or inactive, the device notifies the registered agent. The device can have several instances of this kind of trap to monitor various network types (e.g. WiFi, WCDMA, LTE etc). Call drop trap may be used for bearer selection based on data session drops which occur in the predefined period. Similarly, QoS trap may be used for bearer selection based on received QoS at the device side. Different access technologies have different QoS parameters that maybe monitored. Data rate trap may be used for bearer selection procedure to optimize the device's data rate.

OMA traps are defined as management objects. Each trap management object has unique identifier and a tree structure that allows manipulation of its parameters.

The connectivity management control logic can query the device about the connectivity parameters, i.e. the used network bearer, available network bearers, signal strength as well as network identities. Following preliminary defined policies, the connectivity management logic may decide on the most appropriate bearer to be used, based on diagnostics and monitoring information received by any of the above described traps.

Due to the complexity of device connectivity management, agent technology may be used.

4.2 Agent technology for device connectivity management

An agent is a thing that perceives from and acts on an M2M device in such way that the device goes through a sequence of states maximizing the performance measures. The problem in M2M device connectivity management includes a goal and set of means to achieve the goal. The goal is for the device to use the most appropriate network bearer based on policies. The Connectivity Management Agent reasons about and follows actions in order to achieve the goal. The process of reasoning what means it can do is called search. The Connectivity Management Agent is goal-based and solves the problem deciding what to do by finding sequences of actions that lead to the desirable operational state of M2M devices with cellular or wireless connectivity. The agent actions can be viewed as transitions between M2M device states.

The problem solving of an M2M Connectivity Management Agent includes four stages: goal formulation, problem formulation, searching solution and execution. On receiving a diagnostics and monitoring trap, the Agent explores the current situation and draws the goal which helps to organize behaviour by rejecting actions that result in a failure to achieve the operational state of the M2M device. The Agent draws the problem by deciding what transitions and states to consider following the operational state of M2M device. In general, an M2M Connectivity Management Agent faces with several options of possible sequences of actions because it does not know enough about the current device state. For example, there may be different reasons for device not answering (a connectivity problem, low battery level, a firmware failure, etc.). The Agent searches the solution space by examining different sequences of action. Once the solution is found, the agent carries out the identified actions in the execution stage.

The Connectivity Management Agent in a role of LWM2M server is responsible for observation of device connectivity parameters and selection of best bearer for the device. We assume that the devices' operator has determined preferred bearers for both specific and normal areas. Each device supports traps, which means that the device is capable of monitoring the event and sending notifications whenever it detects the event. The Connectivity Management Agent has to register for the capability in order to use it.

One of the policies of choosing the best network bearer may be based on device location. If the device is in a specified area and the signal strength of the preferred bearer in area is higher than the specified value of TrapActivePower, then the best bearer is the preferred one for this area. When the device is out of the specified area and the signal strength of the preferred bearer out of area is higher than the specified value of TrapActivePower, then the best bearer is the preferred out of area one. If the signal strength of the preferred bearer is lower than the specified value of TrapActivePower, then the best bearer is the available bearer with highest signal strength.

The logic behind the Connectivity Management Agent behaviour might be described as a temporal sequence. On successful device registration, the agent configures geographic traps and received power traps. The agent queries the device about its location and about connectivity parameters. Based on the location, the signal strength of the used network bearer and available bearers, and the best bearer policy the agent performs a bearer selection procedure for the device. After selecting the best bearer the device is in operational state. During this state, the device may send notifications about traps in case of occurrence of the respective event and the agent performs the bearer selection procedure.

Fig.3 shows a simplified model of Connectivity Management Agent where the bearer selection logic is based on device location and received power at the device side. Cognitive behaviour is required when the signal strength of the used bearer is bad but there are no available other bearers.

4.3 Knowledge-base model for device connectivity management

The device and agent have a client-server relationship. We use predicates to express the facts, to show the exchange of messages between the client and server, and to describe the device states as seen by the agent.

Excellent(b, x) becomes true when the received signal strength of bearer b by the device x is higher than the specified value of TrapInactivePower.

Good(b, x) becomes true when the received signal strength of bearer b by the device x is between the specified values of TrapActivePower and TrapInactivePower.

If the device senses signal strength of b below the specified value of TrapActivePower then the Bad(b, x) gets true.

In case the device x uses bearer b then Used(b, x) is true.

InArea(a, x) is true when the device x is in the area a.

Predicates PreferredIn(b, a) and PreferredOut(b, a) are true when bearer b is preferred bearer in area a and out of area a respectively.

The express the fact that bearer b is available for device x the Available(b, x) is used.

When there are no available bearers for x except the used one then AvailableEmpty(x) is true.

BadPreferred(x) is true the received signal strength of preferred bearer by the device x is bad.

Best(b, x) is true if the received signal strength of b is the maximal one for device x.

PowerTrapActive(x, b) gets true when the power trap goes active and the signal strength of used bearer b by device x becomes bad.

PowerTrapInactive(x, b) gets true when the power trap goes inactive and the signal strength of used bearer b by device x becomes excellent.

The behaviour of the Connectivity Management Agent is described by temporal logic. We use a minimal set of standard notations $\mathcal{G}$ for always, $\mathcal{U}$ for until, and $\mathcal{N}$ for next.

The agent considers the following statement when explores the current device state, formulates the problem, searches the solution and performs actions.

The device x is unregistered until a registration request is received:

$$\mathcal{G}(\mathrm{Unregistered}(x)\rightarrow\top\,\mathcal{U}\,\mathrm{reg}_{\mathrm{req}}(x)) \quad (26)$$

After successful device registration, the agent configures geo trap and power trap.

If the device x is unregistered and a registration request is received then a registration response is sent, and a request for geo trap configuration is sent, and the state becomes WaitGeoAck:

$$\mathcal{G}(\mathrm{Unregistered}(x)\wedge\mathrm{reg}_{\mathrm{req}}(x)\rightarrow\mathrm{reg}_{\mathrm{res}}(x)\wedge \neg\mathrm{BadPreferred}(x)\wedge\mathrm{configGeoTrap}_{\mathrm{res}}(x)\wedge \mathcal{N}\mathrm{WaitGeoAck}(x)) \quad (27)$$

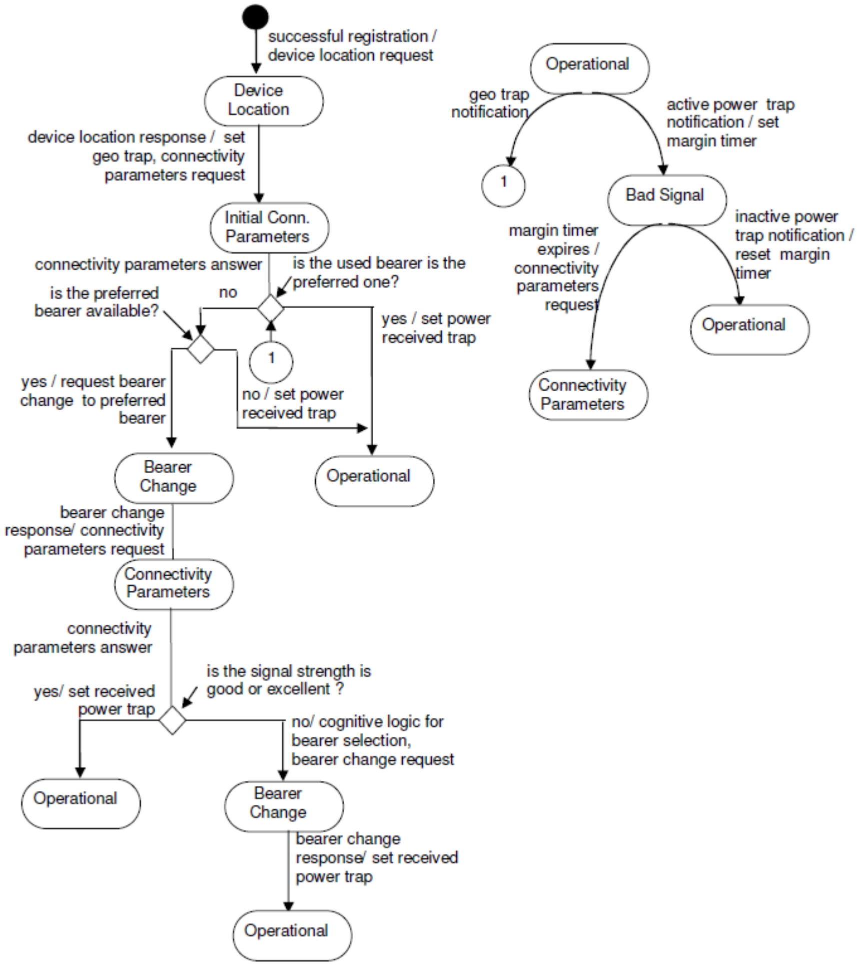

successful registration / device location request
Device Location
device location response / set geo trap, connectivity parameters request
Initial Conn. Parameters
connectivity parameters answer
is the used bearer is the preferred one?
is the preferred bearer available?
no
yes / set power received trap
1
yes / request bearer change to preferred bearer
no / set power received trap
Bearer Change
Operational
bearer change response/ connectivity parameters request
Connectivity Parameters
connectivity parameters answer
is the signal strength is good or excellent ?
yes/ set received power trap
no/ cognitive logic for bearer selection, bearer change request
Operational
Bearer Change
bearer change response/ set received power trap
Operational
Operational
geo trap notification
1
active power trap notification / set margin timer
Bad Signal
margin timer expires / connectivity parameters request
inactive power trap notification / reset margin timer
Operational
Connectivity Parameters

Fig.3 A simplified model of Connectivity Management Agent (bearer selection is driven by device location and received power at the device side)

The device x is in WaitGeoAck state until the agent receives a response of geo trap configuration:

$$\mathcal{G}(\text{WaitGeoAck}(x)\rightarrow\top\,\mathcal{U}\ \text{configGeoTrap}_{\text{res}}(x)) \quad (28)$$

If the state is WaitGeoAck and a response of geo trap configuration is received then a request for configuration of power trap is sent and the state becomes WaitPowerAck:

$$\mathcal{G}(\text{WaitGeoAck}(x)\wedge\text{configGeoTrap}_{\text{res}}(x)\rightarrow$$
$$\text{configPowerTrap}_{\text{req}}(\text{x})\wedge\ \mathcal{N}\text{WaitPowerAck}(x)) \quad (29)$$

The device x is in WaitPowerAck state until the agent receives a response of power trap configuration:

$$\mathcal{G}(\text{WaitPowerAck}(x)\rightarrow$$
$$\top\,\mathcal{U}\text{configPowerTrap}_{\text{res}}(x)) \quad (30)$$

After successful configuration of geo and power traps the agent requests the device location and the device connectivity parameters.

$$\mathcal{G}(\text{WaitPowerAck}(x)\wedge\text{configPowerTrap}_{\text{res}}(x)\rightarrow$$
$$\text{getLocation}_{\text{req}}(\text{x})\wedge\mathcal{N}\text{WaitLocation}(\text{x})) \quad (31)$$

The device x is in WaitLocation state until a location response is received.

$$\mathcal{G}(\text{WaitLocation}(x)\rightarrow\top\,\mathcal{U}\ \text{getLocation}_{\text{res}}(x)) \quad (32)$$

The location response will allow the agent to determine whether the device x is in are a.

$$\mathcal{G}(\text{WaitLocation}(x)\wedge\text{getLocation}_{\text{res}}(x)\rightarrow$$
$$\text{connParameters}_{\text{req}}(x)\wedge\mathcal{N}\text{WaitConnectivity}(x)) \quad (33)$$

The device x is in WaitConnectivity state until a connectivity parameters response is received. The connectivity parameters response will contain the signal strength of used bearer b by device x and available bearers for device x.

$$\mathcal{G}(\text{WaitConnectivity}(x)\rightarrow$$
$$\top\,\mathcal{U}\text{connParameters}_{\text{res}}(x)) \quad (34)$$

Equations from (35) to (38) refer to bearer selection procedure when the device x is in area a.

When the used bearer of *x* is *b*, and *b* is the preferred bearer in area *a*, and the signal strength of *b* is excellent or good, then the state becomes Operational:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \mathrm{InArea}(a, x)\wedge \mathrm{PreferredIn}(b, a)\wedge \mathrm{Used}(b, x) \wedge(\mathrm{Excellent}(b, x)\vee \mathrm{Good}(b, x))\rightarrow \mathcal{N}\mathrm{Operational}(x)) \tag{35}$$

When the used bearer *b* of *x* is the preferred one, and the signal strength of *b* is bad, and *c* is available bearer for device *x* and *c* is the best bearer then a request to select bearer *c* is sent, and the state becomes WaitBearerAck:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \mathrm{InArea}(a,x)\wedge \mathrm{PreferredIn}(b,a)\wedge \mathrm{Used}(b,x)\wedge \mathrm{Bad}(b,x) \wedge \mathrm{Available}(c, x)\wedge \mathrm{Best}(c, a) \rightarrow \mathrm{select}_{\mathrm{req}}(x,c)\wedge \mathrm{BadPreferred}(x)\wedge \mathcal{N}\mathrm{WaitBearerAck}(x)) \tag{36}$$

The bearer *c* selection procedure takes place when *c* is available and preferred bearer, and the received signal strength of *c* by device *x* is not bad:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \mathrm{InArea}(a,x)\wedge \mathrm{Used}(b,x)\wedge \neg \mathrm{PreferredIn}(b,a)\wedge \mathrm{Available}(c,x)\wedge \mathrm{PreferredIn}(c, a)\wedge \neg \mathrm{BadPreferred}(x)\rightarrow \mathrm{select}_{\mathrm{req}}(x, c)\wedge \mathcal{N}\mathrm{WaitBearerAck}(x)) \tag{37}$$

When the used bearer of *x* is not the preferred one, and *c* is available and preferred bearer in area *a*, and the received signal strength of *c* by device *x* is bad, and *d* is the best available bearer then the agent initiates bearer *d* selection procedure:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \mathrm{InArea}(a, x)\wedge \mathrm{Used}(b,x)\wedge \neg \mathrm{PreferredIn}(b,a)\wedge \mathrm{Available}(c,x)\wedge \mathrm{PreferredIn}(c,a)\wedge \mathrm{BadPreferred}(x)\wedge \mathrm{Best}(d,a)\wedge \mathrm{Available}(d,\mathrm{x}) \rightarrow \mathrm{select}_{\mathrm{req}}(x,d)\wedge \mathcal{N}\mathrm{WaitBearerAck}(x)) \tag{38}$$

Equations from (39) to (42) refer to bearer selection procedure when the device *x* is out of area *a*.

When the used bearer *b* of *x* is the preferred one, and the signal strength of *b* is excellent or good, then the state becomes Operational:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \neg \mathrm{InArea}(a, x)\wedge \mathrm{PreferredOut}(b, a) \wedge \mathrm{Used}(b, x)\wedge (\mathrm{Excellent}(b, x)\vee \mathrm{Good}(\mathrm{b}, \mathrm{x})) \rightarrow \mathcal{N}\mathrm{Operational}(x)) \tag{39}$$

When the used bearer *b* of *x* is the preferred one, and the signal strength of *b* is bad, and *c* is the best available bearer then a bearer *c* selection procedure take place:

$$\mathcal{G} (\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \neg \mathrm{InArea}(a,x)\wedge \mathrm{PreferredOut}(b,a)\wedge \mathrm{Used}(b,x)\wedge \mathrm{Bad}(b,x)\wedge \mathrm{Available}(c,x)\wedge \mathrm{Best}(c,a) \rightarrow \mathrm{select}_{\mathrm{req}}(x,c)\wedge \mathrm{BadPreferred}(x)\wedge \mathcal{N}\mathrm{WaitBearerAck}(x)) \tag{40}$$

In case the used bearer *b* of *x* is not the preferred one, and *c* is available preferred bearer, and the received signal strength of *c* by device *x* is not bad, then the agent request selection of bearer *c*:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \neg \mathrm{InArea}(a,x)\wedge \mathrm{Used}(b,x)\wedge \neg \mathrm{PreferredOut}(b,a)\wedge \mathrm{Available}(c, x)\wedge \mathrm{PreferredIn}(c, a)\wedge \neg \mathrm{BadPreferred}(x)\rightarrow \mathrm{select}_{\mathrm{req}}(x,c)\wedge \mathcal{N}\mathrm{WaitBearerAck}(x)) \tag{41}$$

When the signal strength for the preferred bearer *c* is bad and *d* is the best available bearer then the agent request selection of bearer *d*:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x) \wedge \neg \mathrm{InArea}(a,x)\wedge \mathrm{Used}(b,x)\wedge \neg \mathrm{PreferredOut}(b,a) \wedge \mathrm{Available}(c,x)\wedge \mathrm{PreferredIn}(c, a)\wedge \mathrm{BadPreferred}(x)\wedge \mathrm{Best}(d,\mathrm{a})\wedge \mathrm{Available}(d,x) \rightarrow \mathrm{select}_{\mathrm{req}}(x, d)\wedge \mathcal{N}\mathrm{WaitBearerAck}(x)) \tag{42}$$

When there are no available bearers for device *x*, and the received signal strength of *b* by device *x* is good the state becomes Operational:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \mathrm{Used}(b,x) \wedge(\mathrm{Excellent}(b, x)\vee \mathrm{Good}(b, x))\wedge \mathrm{AvailableEmpty}(x)\rightarrow \mathcal{N}\mathrm{Operational}(x)) \tag{43}$$

The agent considers the device *x* unregistered when the received signal strength of the used bearer *b* is bad and there are no available bearers:

$$\mathcal{G}(\mathrm{WaitConnectivity}(x)\wedge \mathrm{connParameters}_{\mathrm{res}}(x)\wedge \mathrm{Used}(b,x)\wedge \mathrm{Bad}(b,x)\wedge \mathrm{AvailableEmpty}(x)\rightarrow \mathcal{N}\mathrm{Unregistered}(x)) \tag{44}$$

The device *x* is in WaitBearerAck state until the agent receives a response of bearer selection procedure:

$$\mathcal{G}(\mathrm{WaitBearerAck}(x)\rightarrow \top\, \mathcal{U}\mathrm{selectBearer}_{\mathrm{req}}(x, b)) \tag{45}$$

When a beater selection response is received, the agent waits for device re-registration:

$$\mathcal{G}(\mathrm{WaitBearerAck}(x)\wedge \mathrm{selectBearers}_{\mathrm{res}}(x)\rightarrow \mathcal{N}\mathrm{WaitReregistration}(x)) \tag{46}$$

The device *x* is in WaitReregistration state until the agent receives a registration request:

$$\mathcal{G}(\mathrm{WaitReregistration}(x)\rightarrow \top\, \mathcal{U}\mathrm{reg}_{\mathrm{req}}(x)) \tag{47}$$

Upon successful device re-registration the agent request device location:

$$\mathcal{G}(\mathrm{WaitReregistration}(x)\wedge \mathrm{reg}_{\mathrm{req}}(x)\rightarrow \mathrm{reg}_{\mathrm{res}}(\mathrm{x})\wedge \neg \mathrm{BadPreferred}(x)\wedge \mathrm{getLocation}_{\mathrm{req}}(x)\wedge \mathcal{N}\mathrm{WaitLocation}(x)) \tag{48}$$

The device *x* is in Operational state until the agent receives a notification about signal strength change of used bearer by x or a notification about change of x location:

$$\mathcal{G}(\mathrm{Operational}(x)\rightarrow \top\,\mathcal{U}\,(\mathrm{notifyPower}_{\mathrm{req}}(x)\vee \mathrm{notifyGeo}_{\mathrm{req}}(x))) \tag{49}$$

In Operational state, when a notification about location change of device x is received, the agent sends a response of geo trap notification and requests device connectivity parameters:

$$\mathcal{G}(\mathrm{Operational}(x)\wedge \mathrm{notifyGeo}_{\mathrm{req}}(x)\rightarrow \mathrm{notifyGeo}_{\mathrm{res}}(x)\wedge \mathrm{connParameters}_{\mathrm{req}}(x)\wedge \mathcal{N}\mathrm{WaitConnectivity}(x)) \tag{50}$$

When the power trap becomes inactive in Operational state, the agent sends a response of power trap notification and the state remains Operational:

$$\mathcal{G}(\mathrm{Operational}(x)\wedge \mathrm{notifyPower}_{\mathrm{req}}(x,b)\wedge \mathrm{PowerTrapInactive}(x)\rightarrow \mathrm{notifyPower}_{\mathrm{res}}(x,b)\wedge \mathcal{N}\mathrm{Operational}(x)) \tag{51}$$

Activation of power trap in Operational state means that the signal strength becomes bad and the agent sets the margin timer:

$$\mathcal{G}(\mathrm{Operational}(x)\wedge \mathrm{notifyPower}_{\mathrm{req}}(x,b)\wedge \mathrm{PowerTrapActive}(\mathrm{x})\rightarrow \mathrm{notifyPower}_{\mathrm{res}}(x,b)\wedge \mathrm{setTmargin}(x)\wedge \mathcal{N}\mathrm{WaitMargin}(x)) \tag{52}$$

The device x is in WaitMargin state until the agent receives a notification about signal strength change of used bearer by x or a notification about change of x location:

$$\mathcal{G}(\mathrm{WaitMargin}(x)\rightarrow \top\,\mathcal{U}(\mathrm{notifyPower}_{\mathrm{req}}(x,b)\vee \mathrm{notifyGeo}_{\mathrm{req}}(x)\vee \mathrm{Tmargin}(x))) \tag{53}$$

Notification that the power trap is inactive in WaitMargin state means that the signal strength becomes excellent and the agent and resets the margin timer:

$$\mathcal{G}(\mathrm{WaitMargin}(x)\wedge \mathrm{notifyPower}_{\mathrm{req}}(x,b)\wedge \mathrm{PowerTrapInactive}\ (\mathrm{x})\rightarrow \mathrm{notifyPower}_{\mathrm{res}}(x,b)\wedge \mathrm{resetTmargin}(x)\wedge \mathcal{N}\mathrm{Operational}(x)) \tag{54}$$

In WaitMargin, when the margin timer expires, the agent sends requests device connectivity parameters:

$$\mathcal{G}(\mathrm{WaitMargin}(x)\wedge \mathrm{Tmargin}(x)\ \rightarrow \mathrm{connParameters}_{\mathrm{req}}(x)\wedge \mathcal{N}\mathrm{WaitConnectivity}(x)) \tag{55}$$

The device may change its location while it is in WaitMargin state:

$$\mathcal{G}(\mathrm{WaitMargin}(x)\wedge \mathrm{notifyGeo}_{\mathrm{req}}(x)\rightarrow \mathrm{notifyGeo}_{\mathrm{res}}(x)\wedge \mathcal{N}\mathrm{WaitMargin}(x)) \tag{56}$$

When the device is unregistered due to connectivity problems, it may wait for some time and try to register again.

5 Model validation

In order to validate the models, we defined OMA management objects related to device connectivity management as resources in REST architecture, following the ETSI approach [29].

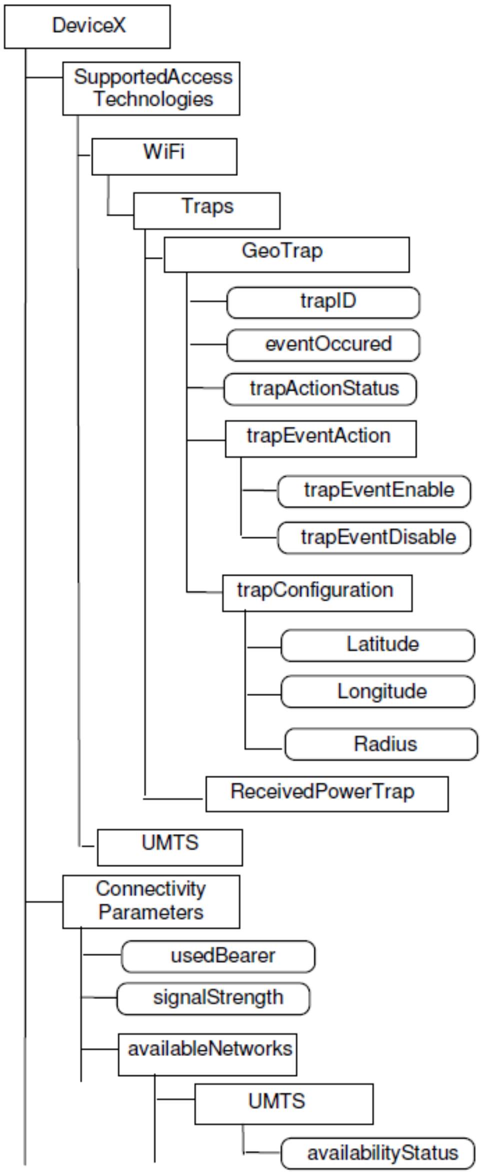


Fig.4 A part of simplified structure of device connectivity management object

Representational State Transfer (REST) is an architectural style that applies principles of distributed systems for loose coupling of components and stateless interactions. In REST, a distributed application (e.g. for connectivity management) is composed of resources, which are stateful pieces of information residing on one or more servers. Resource manipulation is through a uniform interface that is composed of four basic interactions: CREATE, READ, UPDATE and DELETE. The most common implementation of REST is HTTP, whereby the REST primitives are mapped onto HTTP methods, HTTP POST, HTTP GET, HTTP PUT and HTTP DELETE respectively.

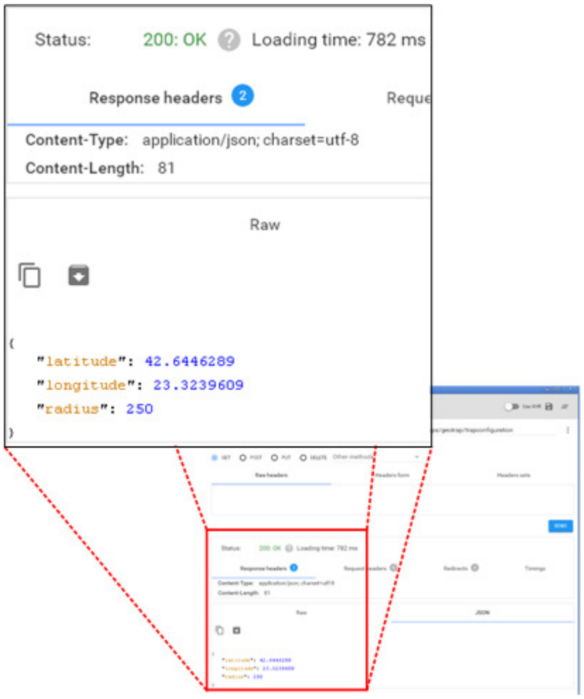
We defined resources representing the device connectivity management objects at the client and server sides. Fig.4 shows a part of the resource structure. The structure is simplified.

The validation process is based on a suit of unit tests that allow comparing the expected message exchange traces to the observed ones.

For illustrative purposes, the Google's Advanced REST client [30] is used in order to depict two basic operations READ and UPDATE.

Fig.5 shows the GET request about query geo trap configuration for Wi-Fi access technology and the result is in JSON format.

Fig.6 shows the HTTP PUT method for updating the geo trap parameters and the respective response.



Status: 200: OK Loading time: 782 ms

Response headers 2 Reque

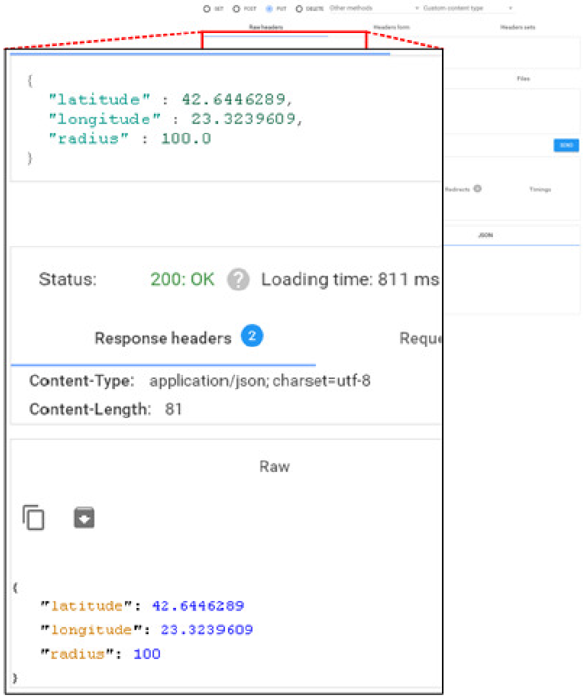
Content-Type: application/json; charset=utf-8

Content-Length: 81

Raw

```
{
 "latitude": 42.6446289
 "longitude": 23.3239609
 "radius": 250
}
```

Fig.5 HTTP GET request for query geo trap configuration parameters



```
{
  "latitude" : 42.6446289,
  "longitude" : 23.3239609,
  "radius" : 100.0
}
```

Status: 200: OK Loading time: 811 ms

Response headers 2 Requ

Content-Type: application/json; charset=utf-8

Content-Length: 81

Raw

```
{
 "latitude": 42.6446289
 "longitude": 23.3239609
 "radius": 100
}
```

Fig.6 HTTP PUT request for update of geo trap configuration parameters

6 Conclusion

Automation of procedures related to M2M device connectivity management reduces operational costs.

The basic paper contribution is the proposition of device connectivity management model, which may be shared by different applications. We model functions for optimization of bearer selection procedure. These functions may be exposed to applications through a set of open interfaces. The model is compliant to OMA LWM2M device management framework. It is based on device capabilities to provide connectivity information such as supported access technologies, used bearer, signal strength, device's location. The model reflects both client (device) and server (cloud) views on connectivity management. It is formally verified using the mathematical methods of bi-simulation. The model is expanded with features that allow designing of autonomous agent. The agent follows a goal related to device connectivity optimization, draws a problem on occurrence of monitoring events and reasons on appropriate actions that have to be executed.

Our future work will include study on service interaction in the context of M2M device management. While the service interaction problem is thoroughly studied for telecommunication services, there is a lack of enough knowledge on the kind of service interactions that occur in the world of M2M communications. Undesired service interaction manifests itself as a function of services which is neither exactly the sum of every service nor behaves as expected. Autonomic resolution of service interactions during service execution is critical task for service continuity.

References:

[1] C. Pereira, A. Aguiar, Towards Efficient Mobile M2M Communications: Survey and Open Challenges, *Sensors* no. 14, 19582-19608; 2014, pp.19582-19608.

[2] J. Holler, V. Tsiatsis, C. Mulligan, S. Avesand, S. Karnouskos, D. Boyle, IoT Architecture – State of the Art, In edited book *From Machine-to-Machine to the Internet of Things: Introduction to a New Age*, Elsevier, 2014, pp.145-165.

[3] M. Elkhodr, S. Shahrestani, Hon Cheung, The Internet of Things: New Interoperability, Management and Security Challenges, *International Journal of Network Security & Its Applications (IJNSA)*,vol.8, No.2, 2016.

[4] H. Park, H Kim, H Joo, J.S. Song, Recent advancements in the Internet-of-Things related standards: A oneM2M perspective, ICT Express, Special Issue on ICT Convergence in the Internet of Things (IoT), vol.2, issue 3, 2019, pp.126-129.

[5] G. Klas, F. Rodermund, Z. Shelby, S. Akhouri, J. Höller, Lightweight M2M: Enabling Device Management and Applications for the Internet of Things, 2014, Available at: http://archive.ericsson.net/service/internet/picov/get?DocNo=1/28701-FGB101973.

[6] Open Mobile Alliance, Enabler Test Specification for Lightweight M2M Candidate Version 1.0 – 03 Feb 2015, OMA-ETS-LightweightM2M-V1_0-20150203-C

[7] J. Sachs, N. Beijar, P. Elmdahl, J. Melen, F. Militano, P. Salmela, Capillary networks – a smart way to get things connected, *Ericsson Review*, no. 8, 2014, pp. 2-8.

[8] A. Sehgal, V. Perelman, S. Kuryla, J. Schönwälder. Management of Resource Constrained Devices in the Internet of Things" *IEEE Communications Magazine*, December 2012, pp.144-149.

[9] Z. Sheng, H. Wang, C. Yin, X. Hu, S. Yang, V. Leung, Lightweight Management of Resource-Constrained Sensor Devices in Internet of Things, *Internet of Things Journal*, Vol.2, Issue 5, 2015, pp.402-411.

[10] D. Schulz, R. Gitzel, Seamless maintenance - Integration of FDI Device Management & CMMS, *IEEE Conference on Emerging Technologies & Factory Automation* (ETFA), 2013, pp.402-407.

[11] C. S. Shih. C. T. Chou, K. J. Lin, B. L. Tsai, C. H Lee, D. Cheng, C. J. Chou, Out-of-Box Device Management for Large Scale Cyber-Physical Systems, *IEEE International Conference on Internet of Things* (iThings), *and Green Computing and Communications* (GreenCom), *and Cyber, Physical and Social Computing* (CPSCom), 2014, pp.402 – 407.

[12] V. Cackovic, Z. Popovic, Device Connection Platform for M2M communications, *IEEE International Conference on Software, Telecommunications and Computer Networks* (SoftCOM), 2012, pp.1-7.

[13] S. Datta, C. Bonnet, Smart M2M Gateway Based Architecture for M2M Device and Endpoint Management, *IEEE International Conference on Internet of Things* (iThings), *and Green Computing and Communications* (GreenCom), *and Cyber, Physical and Social Computing* (CPSCom), 2014, pp.61-68.

[14] A. A. Corici, R. Shrestha, G. Carella, A. Elmangoush, R. Steinke, T. Magedanz, A solution for provisioning reliable M2M infrastructures using SDN and device management, *International Conference on Information and Communication Technology* (ICoICT), 2015, pp.81-86.

[15] E. J. Kim, S. Youm, Machine-to-machine platform architecture for horizontal service integration, *EURASIP Journal on Wireless Communications and Networking*, 2013, doi:10.1186/1687-1499-2013-79, Available at: http://jwcn.eurasipjournals.com/content/2013/1/79

[16] T. Sakamoto and K. Nimura, "Dynamic connection management between Web apps and peripheral devices by Web driver," *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, Sydney, NSW, 2016, pp. 1-6.

[17] D. Kyriazisa, T. Varvarigoua, Smart, autonomous and reliable Internet of Things, *International Workshop on Communications and Sensor Networks* (ComSense'2013), *International Workshop on Communications and Sensor Networks*, ComSense'2013, *Procedia Computer Science*, 2013, pp. 442 – 448.

[18] S. Vassaki, G. Pitsiladis, C. Kourogiorgas, M. Poulakis, A. Panagopoulos, G. Gardikis, S. Costicoglou, Satellite-based sensor networks: M2M sensor communications and connectivity analysis, *International Conference on Telecommunications and Multimedia* (TEMU), Greece, 2014, pp.132–137.

[19] K. Misura, M. Zagar, Internet of things cloud mediator platform, *International Convention on Information and Communication Technology, Electronics and Microelectronics* (MIPRO), 2014, pp.1052-1056

[20] M. Ruta, F. Scioscia, G. Loseto, E. Di Sciascio, Semantic-Based Resource Discovery and Orchestration in Home and Building Automation: A Multi-Agent Approach, *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, 2014, pp.730-741.

[21] S. Frey, A. Diaconescu, D. Menga1, I. Demeure, Towards a generic architecture and methodology for multi-goal, highly-distributed and dynamic autonomic systems, *International Conference on Autonomic Computing* (ICAC), 2013, pp.201-212.

[22] Y. Wang, Formal Cognitive Models of Data, Information, Knowledge, and Intelligence, *WSEAS Transactions on Computers*, 2015, Vol.14, pp.770-781.

[23] G. Cabodi, P. Camurati, C. Loiacono, G. Pipitone, F. Savarese, D. Vendraminetto, Formal Verification of Embedded Systems for Remote Attestation, *WSEAS Transactions on Computers*, 2015, Vol.14, pp.760-769.

[24] G. D'Angelo, S. Ferretti, V. Ghini, Simulation of the Internet of Things. *Proceedings of the IEEE 2016 International Conference on High Performance Computing and Simulation (HPCS 2016)*", pp1-8

[25] Qazi Mamoon Ashraf, Mohamed Hadi Habaebi, Md. Rafiqul Islam, TOPSIS-Based Service Arbitration for Autonomic Internet of Things, *IEEE Access*, vol.4, 2016, pp.1313-1320.

[26] L. Fuchun, Z. Qiansheng, C. Xuesong, Bisimilarity control of decentralized nondeterministic discrete-event systems, *International Control Conference* CCC, 2014, pp.3898-3903.

[27] Open Mobile Alliance (2009). Diagnostics and Monitoring management Object, OMA-TS-DiagMonTrapMO-V1_0-20090414-C

[28] Open Mobile Alliance (2013). Diagnostics and Monitoring Trap Events Specifications, 2013, OMA-TS-DiagonTrapEvents-V1_2-20131008-A

[29] ETSI TS 102 690 Machine-to-Machine communications (M2M); Functional architecture. v1.1.1, 2011.

[30] Google Advanced REST client, 2016, Available at: https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo