BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **17**, No 2
Sofia • 2017 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2017-0016

An Approach to Self-Configuration of M2M Services

Evelina Pencheva

Faculty of Telecommunications, Technical University of Sofia, 8 Kliment Ohridski Blvd., 1756 Sofia, Bulgaria
E-mail: enp@tu-sofia.bg

Abstract**:** *The increased number of connected devices and Machine-to-Machine* (*M2M*) *applications becomes business and technical challenge for network operators. The complexity of connectivity challenge yields for appropriate connectivity management solutions. The explosion of M2M services may result in undesired service interaction. Despite of the considerable progress in service interaction management, there is a lack of knowledge on the kind of interaction in real M2M communication systems. In this paper, a method for service orchestration between M2M applications which add value to basic device connectivity management is proposed. The automatic resolving of service interaction using policies allows service self-configuration and provisioning of adaptive service continuity to end users.*

Keywords**:** *Machine-to-Machine communications, service orchestration, service interaction, description logic, reasoning.*

1. Introduction

Nowadays we are witnessing a widespread penetration of Machine-to-Machine (M2M) communications and Internet of Things (IoT). By 2020, 95% of the devices will be connected to the network using wireless technology. There are numerous M2M applications in various areas of our lives such as e-health, intelligent systems in transport and energy, and smart homes [1]. With the increase of connected devices and end user requirements for new services, new challenges arise for service providers. Preconditions for the provision of new M2M services and applications are requirements for the associate quality of service and security. The number of service providers is constantly growing and they need to cooperate with each other in service provisioning to end users. Capabilities for cooperation between service providers and for infrastructure management have reached the limits of human capacity.

Adding new services is a challenge, since cooperation between multiple service providers and network operators is difficult and requires considering of specific requirements and constrains of any interested party. Service providers expect provision of services in ubiquitous multilayer infrastructure, including virtualization of resources and transparent management. End users are not interested in technology to provide the service, they want to install the service easily and use it without specific configuration [2, 3].

Seamless service continuity is difficult to achieve due to service interaction problem and static service configuration. Service interaction or feature interaction, manifests itself as a function of services which is neither exactly the sum of every service nor behaves as expected. Currently, services are configured for a set of users and a type of access network [4]. The provision of adaptable service continuity requires analysis of M2M environment in a real time and network reconfiguration. The management plane and control plane shall orchestrate service self-configuration based on service self-description capabilities and self-optimization of different network resources.

In this paper, a method for M2M service orchestration is proposed. The method is illustrated for services related to device connectivity management.

The paper is structured as follows. The next section describes the problem of device connectivity management with use cases of service interactions. Section 3 presents the Open Mobile Alliance (OMA) Lightweight Machine-to-Machine device management framework which is the base for the current research. In Section 4, the proposed basic device connectivity model is presented and described formally using descriptive logic. M2M service models that enhance basic device connectivity management capabilities with additional features are presented in Section 5. The content of Section 6 is formed by algorithmic method for M2M service interaction detection and the respective method for M2M service interaction resolution is described in Section 7. Section 8 presents the related work and highlights the added value of the proposed method for service orchestration. The conclusion discusses some implementation aspects.

2. Problem definition

The large and growing category of smart devices including sensors and actuators requires remote configuration and control capabilities. There exists a plethora of device with diverse characteristics in terms of computing and communication capabilities, data rates and form factor which makes device management a real challenge. Device management includes functions like automated device configuration, over-the-air firmware updates, remote reboots, diagnostics and troubleshooting, security and integrity.

Whatever the content and application domain are, all devices involved in M2M communications require some kind of connectivity. Devices may be connected using cellular bearers such as Global System for Mobile communications (GSM), General Packet Radio Service (GPRS), cdmaOne, CDMA2000, EVolution-Data Optimized (EV-DO), Enhanced Data rates for GSM Evolution (EDGE), Universal Mobile Telecommunications System (UMTS), Digital AMPS (IS-136/TDMA), Integrated Digital Enhanced Network (iDEN), Long Term Evolution (LTE), etc. Devices may also use wireless technologies like Near Field

Communications (NFC), WiFi Direct, WiFi Passpoint, WiGig, WiFi Miracast, Bluetooth, ZigBee, Dash7, EnOcean etc. Different technologies have different requirements for Quality of Service (QoS). Connectivity management encompasses connection provisioning, management and analysis across networks to optimize data consumption and cost. Connectivity management platform should at minimum allow control on the device status and capabilities for choosing the best network bearer to be used. The logic for bearer selection may be based on different policies such as the device location and the requirements for charging.

Let us consider a payment application which controls the usage of network bearers based on the current device provider's credit. The application defines a threshold under which the device provider's balance is considered to be low. When the balance is low the application requires the device to use the cheapest available bearer. Applications like remote control and multiuser gaming require lower latency which is one of the QoS parameters. Such applications may define thresholds related to QoS parameters and request usage of network bearers that fulfil the QoS requirements. A possible service interaction may occur when the payment application requires usage of the cheapest bearer like GSM but the QoS application requires using LTE due to latency requirements.

Another use case scenario is the interaction between location-based application and data speed-based application. Location-based application may define an area in which a preferred bearer has to be used, e.g. a WiFi bearer may be preferred in a home area. Bandwidth hungry applications like video surveillance, transportation services, and industrial control may require high data speeds which are not supported by a particular technology. A possible service interaction may occur when the device uses a network bearer that provides the requested data speeds and enters an area where the preferred bearer does not support the required data speeds.

Having in mind the variety of cellular and wireless technologies, the supported data speeds in uplink and downlink directions, different QoS parameters and other policies for bearer preferences like location-based, credit-based, serving node-based etc., a lot of combination of undesired service interactions may occur. The reduction of device connectivity management complexity can be achieved by embedding autonomic features in M2M service orchestration. The paper proposes a basic device connectivity model and a method for automatic service orchestration between M2M applications which add value to basic bearer selection procedure.

Next section presents the background for the current research.

3. OMA device management framework

OMA device management framework called LightWeight M2M (LWM2M) provides an abstraction of device management which hides the complexity and is technology independent [5]. LWM2M describes remote management procedures between a server in the cloud or network, and a client in the device, and defines the respective information model. 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 Access Point Name profile.

7. The server cancels observation.

The observed connectivity parameters include the used network bearer for the current communication session, the list of current available network bearers, the average value of the received signal strength indication used in the current network bearer, and the received link quality. In order to observe connectivity parameters, OMA specifications define a number of diagnostics and monitoring traps [6, 7]. The trap mechanism employed by a management authority to enable the device to capture and report events and other relevant information generated from various components of the device, such as a protocol stack, device drivers, or applications.

OMA traps that may be used for connectivity management are geographic trap, 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 be 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 sensed power rises above another server specified value (TrapInactivePower), it causes this trap to go inactive. In cases that the trap goes active or inactive, the device notifies the server. The device can have several instances of this kind of trap to monitor various network types (e.g., WiFi, WCDMA, LTE, etc.). The remote server may observe the call drops in a predefined period of time. If the device exposes QoS metrics functionality, then the server may observe the received QoS at the device side using the QoS trap. The data speed trap occurs whenever an average data speed reaches the certain threshold value.

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.

4. Device connectivity management model

The approach to studying of M2M service interaction problem is based on author's previous work, where the problem is explored for CAMEL networks [8-11]. Customized Applications for Mobile Enhanced Logic (CAMEL) is service platform for GSM and UMTS networks. The research was focused on human call-related behavior. In [8] and [9], service interactions based on CAMEL originating and terminating basic call models respectively and reasoned on interactions between services available for calling and called party. In [10, 11], the focus is on CAMEL mobility management models where service interactions are result of subscriber mobility. CAMEL models are not applicable in the world of M2M communications where devices are used for data transfer and the human interactions is lack or limited.

In order to send information over the network, any device needs connectivity. The study on M2M service interaction is based on the basic Device connectivity management model, shown in Fig.1.

In *disconnected* state, the device is not connected to the network and can not communicate. The device moves to *connected* state when it is turned on and after successful registration with the server. In *connected* state, the device uses one of the supported network bearers. The server may configure the received power trap in order to observe the used bearer signal power. When the received power drops, the server is notified and it sets a timer with the time guarded hysteresis of the received power. The device moves to *marginal* state. In *marginal* state, the device moves to *connected* state when the signal received power rises and then the server resets the hysteresis timer. In *marginal* state, when the hysteresis timer expires, the server queries the device about connectivity parameters and the device moves to *badSignal* state. In *badSignal* state, if there are not available network bearers, the server requests the device to disconnect. In *badSignal* state, if there is an available network bearer, the server requests the device to change the used bearer. In any state but *disconnected*, the device may be disconnected due to de-registration initiated by the server or the device.

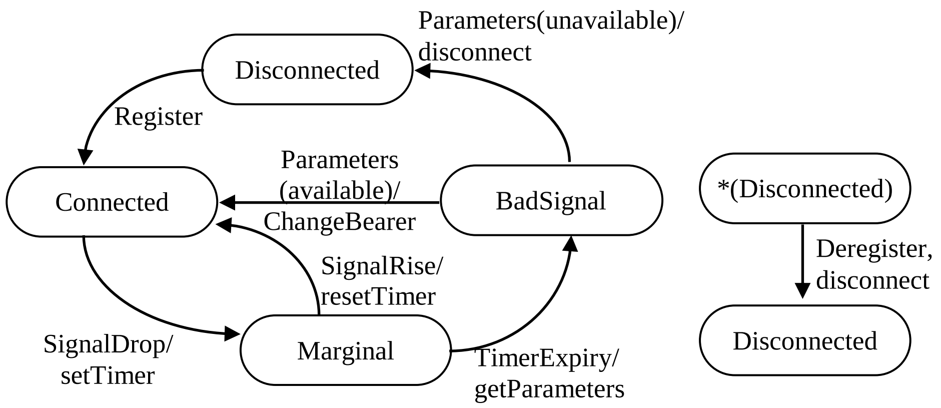

Fig. 1. Basic device connectivity management model

Description logic is used to formally describe the basic device connectivity management model.

First, atomic concepts, relations and constants are fixed. Assume that there are a finite set of devices which consists of devices in the M2M system.

The following concepts express the device state and facts related to the device connectivity:

disconnected – the device is disconnected
connected$_b$ – the device is connected by bearer *b*
marginal$_b$ – the device's received power of bearer *b* is below a server-specified value
badSignal$_b$ – the device needs to change the used bearer *b*
available$_b$ – the bearer *b* is available

Roles represent actions or notifications about events related to device connectivity management.

Register – the device registers to the server
getParameters – the server queries the device about connectivity parameters
parameters – the device provides the requested connectivity parameters
signalDrop – the received power drops below server-specified value (TrapActivePower)
signalRise – the received power rises above server-specified value (TrapInactivePower)
setTimer – the server sets the time guarded hysteresis of the received power
timerExpiry – the time guarded hysteresis of the received power is over
deregister – the device de-registers with the server
disconnect – the server requests the device to disconnect
changeBearer – the server instructs the device to change the used bearer

Concepts and roles are used to specify the Device Connectivity Management Model (CMM). The Terminology Box (TBox) consists of expressions that represent how the device can change its state, and:

(1) $\text{disconnected} \sqsubseteq \exists \text{register}.\text{connected}_b$,

(2) $\text{connected}_b \sqsubseteq \exists \text{getParameters}.\text{connected}_b$,

(3) $\text{connected}_b \sqsubseteq \exists \text{parameters}.\text{connected}_b \sqcap \text{available}_c$,

(4) $\text{connected}_b \sqsubseteq \exists \text{parameters}.\text{connected}_b \sqcap \neg \text{available}_c$,

(5) $\text{connected}_b \sqsubseteq \exists (\text{signalDrop} \sqcap \text{setTimer}).\text{marginal}_b$,

(6) $\text{marginal}_b \sqsubseteq \exists (\text{signalRise} \sqcap \text{resetTimer}).\text{connected}_b$,

(7) $\text{marginal}_b \sqsubseteq \exists (\text{timerExpiry} \sqcap \text{getParameters}).\text{badSignal}_b$,

(8) $\text{badSignal}_b \sqsubseteq \exists \text{parameters}.(\text{badSignal}_b \sqcap \text{available}_c)$,

(9) $\text{badSignal}_b \sqcap \text{available}_c \sqsubseteq \exists \text{changeBearer}.\text{connected}_b$,

(10) $\text{badSignal}_b \sqsubseteq \exists \text{parameters}.(\text{badSignal}_b \sqcap \neg \text{available}_c)$,

(11) $\text{badSignal}_b \sqcap \neg \text{available}_c \sqsubseteq \exists \text{disconnect}.\text{disconnected}$,

(12) $\text{connected}_b \sqsubseteq \exists (\text{disconnect} \sqcup \text{deregister}).\text{disconnected}$,

(13) $\text{marginal}_b \sqsubseteq \exists(\text{disconnect} \sqcup \text{deregister}).\text{disconnected}$,

(14) $\text{badSignal}_b \sqsubseteq \exists(\text{disconnect} \sqcup \text{deregister}).\text{disconnected}$.

The Assertion Box (ABox) contains one statement presenting the initial state for each device:

$$s_0 : \sqcap_{d \in \text{Devices}} \text{disconnected}.$$

To express the fact that each device is in exactly one state at any moment the following statement is used:

$$\top \sqsubseteq \neg(\sqcup_{d_1, d_2 \in \text{CMM},\ d_1 \neq d_2}(s_1 \sqcap s_2)) \sqcap (\sqcup_{d \in \text{CMM}}\, s).$$

The device state changes by means of actions defined as action functions. An action function Func_{CMM} for given state corresponds to the possible transitions in the CMM. For example, the expression $\text{Func}_{\text{CMS}}(\text{connected}_b)$= signalDrop}$\cup${disconnect}$\cup${deregister} means that, if the device is connected, the received power of the used bearer may drop, the device may disconnect or deregister.

The fact that each device can change the CMM state only by means of certain actions is represented by the following statement: for all $s \in \text{CMM}$, and all $R \notin \text{Func}_{\text{CMM}}(s)$, $s \sqsubseteq \forall R.s$.

5. Service models

Services are modelled as transformations on the knowledge base using contexts $C[\varphi]$ as subformula φ of any formula ψ.

5.1. Location-based bearer selection

The Location-based Bearer Selection (LBS) service assumes that there is a predefined geographic area in which a preferred bearer is used. The transformation of the basic device connectivity management model for LBS is shown in Fig. 2.

When the device registers, the server queries the device about its location and connectivity parameters. If the device is in the specified area and it does not use the preferred bearer and the preferred bearer is available, the server requests the device to change the bearer. If the device is out of the area, it may enter the area. If the device is in area, it may exit the area. Additional concepts representing facts are defined:

inArea	the device is located in the specified area
outOfArea	the device is located out of the specified area
preferred_b	the bearer b is the preferred one in the specified area

The following relationship is true: $\text{outOfArea} \equiv \neg\text{inArea}$

enter	the device enters the specified area
exit	the device exits the specified area
location	the device sends its location
getLocation	the server queries about device’s location

The refinement for LBS service is defined by the following statements:

(15) $C_1[\text{LBS}\sqcap\text{disconnected}]\sqsubseteq\exists\text{register}.C_2[\text{connected}_b]$,

(16) $C_3[\text{LBS}\sqcap\text{connected}_b]\sqsubseteq\exists\text{getLocation}.C_4[\text{connected}_b]$,

(17) $C_5[\text{LBS}\sqcap\text{connected}_b]\sqsubseteq\exists\text{location}.C_6[\text{connected}_b\sqcap\text{inArea}]$,

(18) $C_7[\text{LBS}\sqcap\text{connected}_b]\sqsubseteq\exists\text{location}.C_8[\text{connected}_b\sqcap\text{outOfArea}]$,

(19) $C_9[\text{LBS}\sqcap\text{connected}_b\sqcap\text{inArea}]\sqsubseteq$
$\exists\text{parameters}.C_{10}[\text{connected}_b\sqcap\text{inArea}\sqcap\text{preferred}_b]$,

(20) $C_{11}[\text{LBS}\sqcap\text{connected}_b\sqcap\text{inArea}]\sqsubseteq$
$\exists\text{parameters}.C_{12}[\text{connected}_b\sqcap\text{inArea}\sqcap\text{preferred}_c\sqcap\text{available}_c]$,

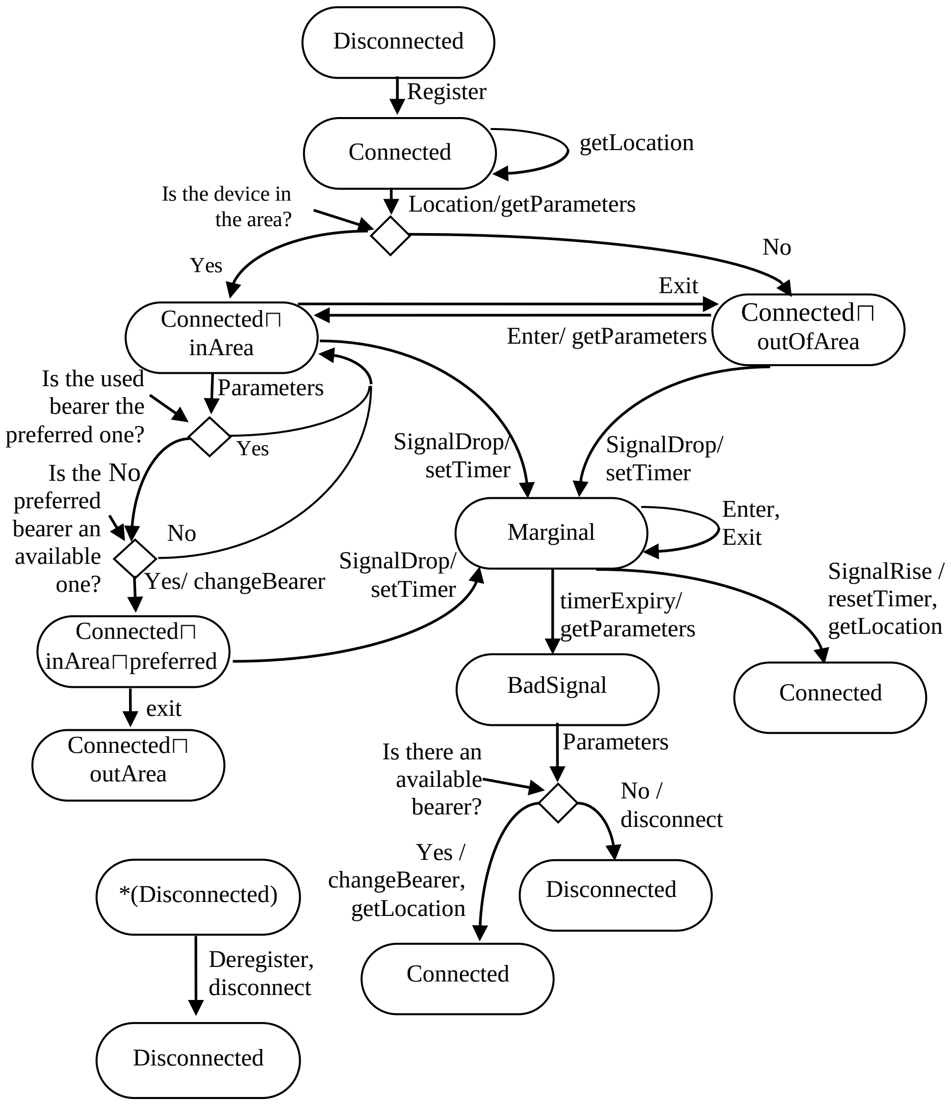

Fig. 2. Location-based bearer selection

(21) $C_{13}[\text{LBS}\sqcap\text{connected}_b\sqcap\text{inArea}\sqcap\text{preferred}_c\sqcap\text{available}_c]\sqsubseteq$
$\exists\text{changeBearer}.\ C_{14}[\text{connected}_c\sqcap\text{inArea}\sqcap\text{preferred}_c]$,

(23) $C_{15}[\text{LBS}\sqcap\text{connected}_b\sqcap\text{inArea}]\sqsubseteq\exists\text{exit}.\ C_{16}[\text{connected}_b\sqcap\text{outOfArea}]$,

(24) $C_{17}[\text{LBS}\sqcap\text{connected}_b\sqcap\text{outOfArea}]\sqsubseteq\exists\text{enter}.C_{18}[\text{connected}_b\sqcap\text{inArea}]$,

(25) $C_{19}[\text{LBS}\sqcap\text{marginal}_b]\sqsubseteq\exists(\text{exit}\sqcup\text{enter}).C_{20}[\text{marginal}_b]$,

(26) $\text{LBS}\sqsubseteq\neg(\text{connected}_b\sqcap\text{inArea}\sqcap\text{preferred}_c\sqcap\text{available}_c)$.

5.2. Credit-based bearer selection

The Credit-based Bearer Selection (CBS) service assumes that the device provider has prepaid subscription. If the device provider's balance is low, the device needs to use the cheapest bearer. Real-time information about device provider's balance may be acquired by means of Policy and Charging Control (PCC) functionality standardized for mobile networks. The PCC concept is designed to enable flow based charging including online credit control and policy control which supports service authorization and quality of service management [12].

The transformation of the basic device connectivity management model for CBS is shown in Fig. 3.

The server queries about the device provider's balance status and device connectivity parameters. If the balance is lower than the server defined threshold and the device does not use the cheapest bearer, and the cheapest bearer is available, the server requests the device to change the bearer to the cheapest one. The device provider's balance may increase above a server-defined threshold or decrease under a server-defined threshold. Additional concepts representing facts are defined as follows:

lowBalance	the device provider's balance is low
highBalance	the device provider's balance is low
cheapest_b	the bearer b is the cheapest one

There exists the following statement in the TBox: $\text{lowBalance}\equiv\neg\text{highBalance}$.
Additional roles representing actions and notifications are also defined.

increase	the device provider's balance increases above a server-defined threshold
decreases	the device provider's balance decreases under a server-defined threshold
balance	the server receives device provider's balance status
getBalance	the server queries about device provider's balance status

The refinement for LBS service is defined by the following statements:

(27) $C_{21}[\text{CBS}\sqcap\text{disconnected}]\sqsubseteq\exists\text{register}.C_{22}[\text{connected}_b]$,

(28) $C_{23}[\text{CBS}\sqcap\ \text{connected}_b]\sqsubseteq\exists\text{getBalance}.C_{24}[\text{connected}_b]$,

(29) $C_{25}[\text{CBS}\sqcap\text{connected}_b]\sqsubseteq\exists\text{balance}.C_{26}[\text{connected}_b\sqcap\text{lowBalance}]$,

(30) $C_{27}[\text{CBS}\sqcap\text{connected}_b]\sqsubseteq\exists\text{balance}.C_{28}[\text{connected}_b\sqcap\text{highBalance}]$,

(31) C_{29}[CBS⊓connected$_b$⊓lowBalance]⊑
∃parameters.C_{30}[connected$_b$⊓lowBalance⊓cheapest$_b$],

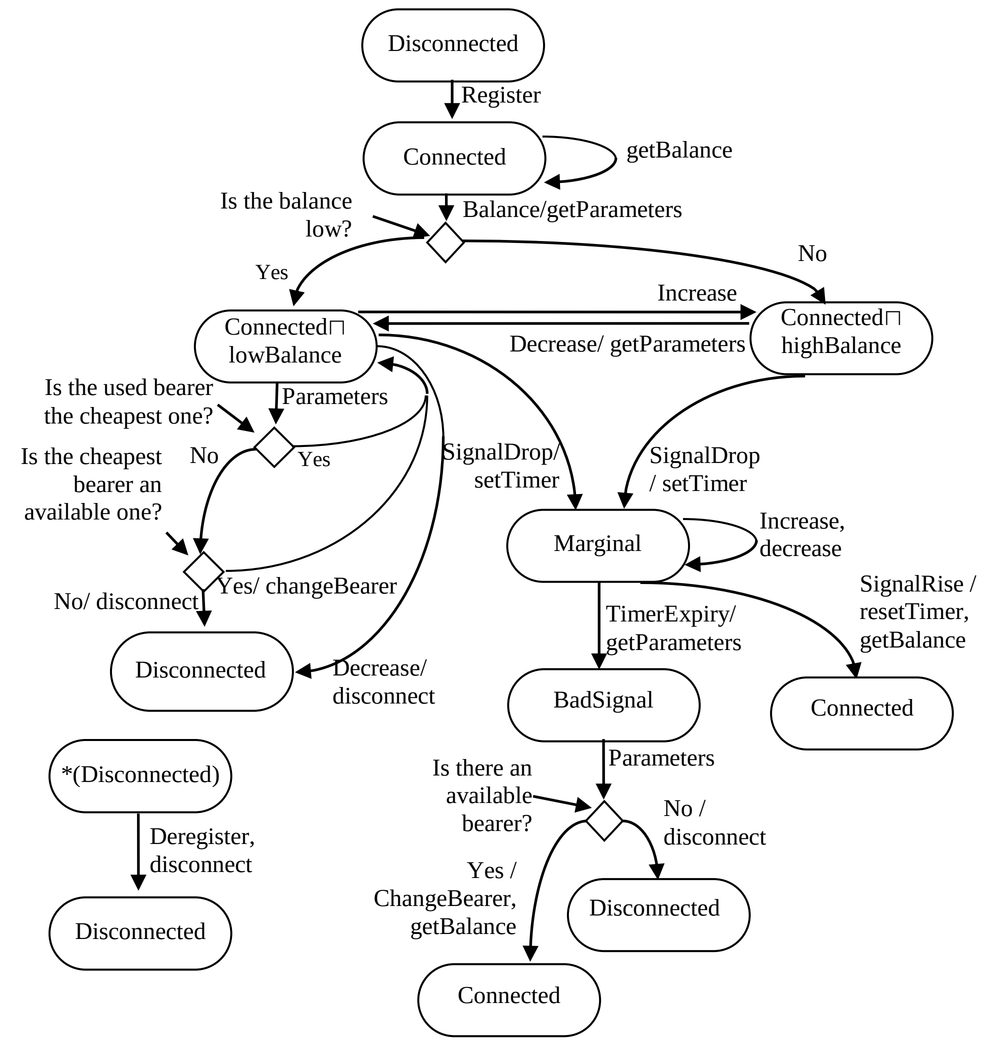

Fig. 3. Credit-based bearer selection

(32) C_{31}[CBS⊓connected$_b$⊓lowBalance]⊑
∃parameters.C_{32}[connected$_b$⊓lowBalance⊓cheapest$_c$⊓available$_c$],

(33) C_{33}[CBS⊓connected$_b$⊓lowBalance⊓cheapest$_c$⊓available$_c$]⊑
∃changeBearer.C_{34}[connected$_c$⊓lowBalance⊓cheapest$_c$],

(34) C_{35}[CBS⊓connected$_b$⊓lowBalance⊓cheapest$_c$⊓¬available$_c$]⊑
∃disconnect.C_{36}[disconnected],

(35) C_{37}[CBS⊓connected$_b$⊓highBalance]⊑
∃descrease.C_{38}[connected$_b$⊓lowBalance],

(36) C_{39}[CBS⊓connected$_b$⊓lowBalance]⊑
∃(descrease⊓disconnect).C_{40}[disconnected],

(37) C_{41}[CBS⊓connected$_b$⊓lowBalance]⊑∃increase.C_{42}[connected$_b$⊓highBalance],

(38) C_{43}[CBS⊓marginal$_b$]⊑∃(decrease⊔increase).C_{44}[marginal$_b$],

(39) CBS⊑ ¬(connected$_b$⊓lowBalance⊓cheapest$_c$⊓available$_c$).

5.3. Other services related to device connectivity management

By the use of OMA Diagnostic and monitoring traps different connectivity management services may be defined.

The QoS bearer selection service may use the QoS trap. The service requires configuration of lower QoS threshold and upper QoS threshold, where the value format must be interpreted according to the parameter definition. The QoS trap is enabled when its value is equal or greater than the specified value of lower QoS threshold, or its value is minor or equal than the value of upper QoS threshold. The knowledge base for QoS bearer selection procedure is expanded with new concepts related to QoS thresholds, new roles reflecting the increase and decrease of the value of QoS parameters, as well as statements that represent the relationship between QoS concepts and roles.

The Call drop bearer selection service may use the call drop trap. The service configures the start and end times of the observation and initiates bearer selection procedure whenever a call drop occurs in a specified period. The call drop trap has to be considered in the context of data session drops for machine type communications. The knowledge base for this service is extended with new concepts, roles related to occurrence of data session drops and statements for bearer selection logic.

Data speed bearer selection service may use the data speed trap. The service configures different data speed traps for uplink and downlink. Low speed data traps become active when the average data speed calculated for the given period reaches below the server defined lower threshold value. High speed data traps become active when the average data speed calculated for the given period reaches above this higher threshold value. The service initiates bearer selection whenever the data speed trap goes to active. The knowledge base for this service is extended with new concepts representing low and high data speed thresholds for both directions, new roles for trap activity and statements for bearer selection logic.

6. Service interaction as satisfiability problem

When introducing new services, it is important to find out whether a new service is contradictory to existing concepts, i.e., whether it satisfies or not the statement in the TBox representing the connectivity management model.

A standard tableau method is used, where the tableau $t \overset{\text{def}}{=} \{ \langle b \mid p\text{: } C \rangle \}$ is a set of prefixed formulae where the prefix of given formula is consisted of a binary string $b := \varepsilon \mid (1|0)^+$ and a string of alternating names $p := n(Rm)^+$, and C is concept. Here ε is the empty string, n and m are individual names, R is role names, and $()^+$ denotes one or more occurrences. The method is shown in Table 1.

Table 1. Tableau method

Conjunction: $\dfrac{\langle b \mid p : C \cap D \rangle}{\langle b \mid p : C \rangle \quad \langle b \mid p : D \rangle}$	
Disjunction: $\dfrac{\langle b \mid p : C \cup D \rangle}{\langle b_M 0 \mid p : C \rangle \quad \langle b_M 1 \mid p : D \rangle}$	b_M maximal for b
Existence: $\dfrac{\langle b \mid p : \exists R.C \rangle}{\langle b \mid pRn : C \rangle}$	pRn new (unless pR exists in the branch)
Implication: $\dfrac{\vdots}{\langle b \mid p : \neg C \cup D \rangle}$	p present in b and $C \sqsubseteq D \in T$

The algorithm for detection of service interaction is illustrated for LBS and CBS services. The service interaction occurs when the device is in the specified area and uses the preferred bearer as to LBS, and the device provider's balance decreases under predefined value and the CBS requires a change to the cheapest bearer.

Let us assume that the device provider is subscribed for both LBS and CBS. Starting with the axiom in the ABox we apply statements in the TBox related to the event sequence that leads to the undesired situation.

Algorithm

Step 1. Initially, the device is disconnected. Applying *Conjunction* to the start formula $\langle \varepsilon \mid s_0\text{: } \sqcap_{d \in \text{Devices}} \text{disconnected} \rangle$ gives

$$\langle \varepsilon \mid s_0\text{: disconnected} \rangle \ ,$$

Step 2. Applying *Implication* to rule (1) produces

$$\langle \varepsilon \mid s_0\text{: } \neg\text{disconnected} \sqcup \exists \text{registered.connected}_b \rangle \ ,$$

Step 3. Applying *Disjunction* gives two branches:

Step 3.1. $\langle 0 \mid s_0\text{: } \neg\text{disconnected} \rangle$ which is closed because of appearance of $\langle \varepsilon \mid s_0\text{: disconnected} \rangle$ in this segment.

Step 3.2. $\langle 1 \mid s_0\text{: } \exists\text{registered.connected}_b \rangle$,

Step 4. Applying *Existence* leads to new state s_1: $\langle 1 \mid s_0\text{: registered } s_1\text{: connected}_b \rangle$,

Step 5. In connected state, the device is queried about its location. We derive rule (16) from the knowledge base. Applying *Implication* and than *Disjunction* produces:

Step 5.1. $\langle 10 \mid s_0{:}\ \neg\text{connected}_b \rangle$ which is closed because of appearance of $\langle 1 \mid s_0{:}$ registered $s_1{:}\ \text{connected}_b \rangle$ in this segment.

Step 5.2. $\langle 11 \mid s_0$ registered $s_1{:}\ \text{connected}_b \sqcup \exists\text{getLocation.connected}_b \rangle$ to which after applying *Existence* produces $\langle 11 \mid s_0{:}$ registered $s_1{:}\ \text{connected}_b$ getLocation $s_1{:}\ \text{connected}_b \rangle$.

Step 6. We consider the case when the device is in the specified area. The next derivation is rule (17). The consecutive application of *Implication, Disjunction* and *Existence* results in:

$\langle 111 \mid s_0$ registered s_1 getLocation s_1 location $s_2{:}\ \text{connected}_b \sqcap \text{inArea} \rangle$.

Step 7. If the device is in the specified area, it is queried about its connectivity parameters. We derive rule (2) and apply *Implication, Disjunction* and *Existence,* which produces:

$\langle 1111 \mid s_0$ registered s_1 getLocation s_1 location s_2 getParameters $s_2{:}$ $\text{connected}_b \sqcap \text{inArea} \rangle$.

Step 8. Let the device uses the preferred bearer in the specified area. We apply *Implication, Disjunction* and *Existence* to rule (19) and the result is:

$\langle 11111 \mid s_0$ registered s_1 getLocation s_1 location s_2 getParameters s_2 parameters $s_3{:}\text{connected}_b \sqcap \text{ inArea} \sqcap \text{preferred}_b \rangle$.

Step 9. Let the device provider's balance initially be high. We derive rules (28) and (30) for which *Implication, Disjunction* and *Existence* are applied. The result is

$\langle 1111111 \mid s_0$ registered s_1 getLocation s_1 location s_2 getParameters s_2 parameters s_3 getBalance s_3 balance $s_4{:}\ \text{connected}_b \sqcap \text{inArea} \sqcap \text{preferred}_b \sqcap \text{highBalance} \rangle$.

Step 10. After some time the device provider's balance decreases under a specified threshold.

Again *Implication, Disjunction* and *Existence* are consecutively applied to rule (35). The result is:

$\langle 11111111 \mid s_0$ registered s_1 getLocation s_1 location s_2 getParameters s_2 parameters s_3 getBalance s_3 balance s_4 decrease $s_5{:}$ $\text{connected}_b \sqcap \text{inArea} \sqcap \text{preferred}_b \sqcap \text{lowBalance} \rangle$.

Step 11. The device is queried about its connectivity parameters. *Implication, Disjunction* and *Existence* are applied again to rules (2) and (32) which leads to:

$\langle 1111111111 \mid s_0$ registered s_1 getLocation s_1 location s_2 getParameters s_2 parameters s_3 getBalance s_3 balance s_4 decrease s_5 getParameters s_5 parameters $s_6{:}$ $\text{connected}_b \sqcap \text{inArea} \sqcap \text{preferred}_b \sqcap \text{lowBalance } \sqcap \text{cheapest}_c \sqcap \text{available}_c \rangle$.

Step 12. When the device provider's balance is low and there is unused cheapest bearer, the CBS service requires a bearer change. We derive rule (33) and after the consecutively application of *Implication, Disjunction* and *Existence* the result is

$\langle 1111111111 |\ s_0$ registered s_1 getLocation s_1 location s_2 getParameters s_2 parameters s_3 getBalance s_3 balance s_4 decrease s_5 getParameters s_5 parameters s_6 changeBearer s_7:$\mathrm{connected}_c \sqcap \mathrm{inArea} \sqcap \mathrm{preferred}_b \sqcap \mathrm{lowBalance} \sqcap \mathrm{cheapest}_c \rangle$ which contradicts to (26) as to LBS, namely

$$\neg(\mathrm{connected}_b \sqcap \mathrm{inArea} \sqcap \mathrm{preferred}_c \sqcap \mathrm{available}_c).$$

The result is closed tableau which means that $\delta_{\mathrm{LBS}}(\delta_{\mathrm{CBS}}(\mathrm{CMM}))$ interacts on activation $\{\mathrm{LBS}\} \cup \{\mathrm{CBS}\}$.

7. Resolving service interaction

Once detected, service interactions may be resolved by policies. The policies define the service behavior in case of service interworking and they are expressed also by contexts.

Different policies may be defined for LBS and CBS interworking, based on service priority. The priority of *i* service is denoted by P_i.

If both services have equal priorities, i.e., $P_{\mathrm{LBS}}=P_{\mathrm{CBS}}$, the policy may request device disconnection in case the cheapest bearer is not the preferred one in the specified area. This is the case when no negotiation between services is possible:

(40) $C_{45}[\mathrm{LBS} \sqcap \mathrm{CBS} \sqcap P_{\mathrm{LBS}}=P_{\mathrm{CBS}} \sqcap \mathrm{connected}_b \sqcap \mathrm{inArea} \sqcap \mathrm{preferred}_b \sqcap \mathrm{lowBalance} \sqcap \mathrm{cheapest}_c \sqcap \mathrm{available}_c] \sqsubseteq \exists \mathrm{disconnect}.C_{46}[\mathrm{disconnected}]$.

If a negotiation between both services is possible, then the policy depends on the higher service priority. In case of $P_{\mathrm{LBS}}<P_{\mathrm{CBS}}$, the policy requires bearer change to the cheapest bearer nevertheless it is not the preferred one in the specified area:

(41) $C_{47}[\mathrm{LBS} \sqcap \mathrm{CBS} \sqcap P_{\mathrm{LBS}}<P_{\mathrm{CBS}} \sqcap \mathrm{connected}_b \sqcap \mathrm{inArea} \sqcap \mathrm{preferred}_b \sqcap \mathrm{lowBalance} \sqcap \mathrm{cheapest}_c \sqcap \mathrm{available}_c] \sqsubseteq \exists \mathrm{changeBearer}.C_{48}[\mathrm{connected}_c \sqcap \mathrm{inArea} \sqcap \mathrm{preferred}_b \sqcap \mathrm{lowBalance} \sqcap \mathrm{cheapest}_c]$.

In case of $P_{\mathrm{LBS}}>P_{\mathrm{CBS}}$, the policy requires the device to use the preferred bearer until the device provider’s balance goes down to zero:

(42) $C_{49}[\mathrm{LBS} \sqcap \mathrm{CBS} \sqcap P_{\mathrm{LBS}}>P_{\mathrm{CBS}} \sqcap \mathrm{connected}_b \sqcap \mathrm{inArea} \sqcap \mathrm{preferred}_b \sqcap \mathrm{lowBalance} \sqcap \mathrm{cheapest}_c \sqcap \mathrm{available}_c] \sqsubseteq C_{50}[\mathrm{connected}_c \sqcap \mathrm{inArea} \sqcap \mathrm{preferred}_b \sqcap \mathrm{lowBalance}]$.

8. Related work

The survey on service orchestration research shows that works consider either high level architectural principles or specific implementations, but do not propose a structural approach to resolving conflict between services.

In [13], the authors argue the usage of ontologies in service interoperability issues presenting ontology for resources and operations. The potential of ontology allows coping with problems in service interaction also as in the proposed approach. Chen and Gian present an orchestration environment VIPLE (Visual IoT/Robotics

Programming Environment), which is an IoT middleware with open interfaces to IoT devices [14]. Im and Jeong propose a new service platform that facilitates the implementation of new applications by composing and orchestration of prebuilt components that provide the context information of mobile devices such as location and contacts [15]. Their platform adopts event-driven architecture to work intelligently with context awareness and do not consider the device connectivity. In [16], the authors propose a service oriented middleware that leverages the convergence of cloud and fog computing along with software defined networking and network function virtualization. They describe a system which abstracts connected entities as services and allows applications to orchestrate these services with end to end QoS requirements. In comparison, the approach proposed in this paper deals also with requirements based on device location and charging. A service architecture which includes various devices for providing web base IoT services is described in [17]. The authors also propose a service platform which supports service orchestration and composition with device objectification. D'Angelo, Ferretti and Ghini claim that agent-based, adaptive parallel and distributed simulation approaches are needed for IoT environments, together with multi-level simulation, which provide means to perform highly detailed simulations, on demand. They present a use case concerned with the simulation of smart territories [18].

Works on IoT service orchestration focus on the process of integrating applications and/or services together to automate a process, or synchronize data in real-time, but do not consider the problem of service interaction which appears to be an essential part of service orchestration problems.

Despite of the progress in developing approaches for modeling, detecting, and resolving service interactions, there is a lack of sufficient knowledge on the kind of service interactions that occur in real-world M2M systems [19]. Instances of the service interaction problem have been studied in different IoT applications like home automation [20], automotive systems [21], systems of services [22] and in many other fields. In [23], the authors present a method to check for feature interactions in a system assembled from independently developed concurrent processes as found in many reactive systems which combines and refines existing definitions and adds a set of activities. The method is illustrated on a home automation example. The compositionality and modularity are considered to be in the base of the problem instances, while the difference between the individual views, interpretations and eventual solutions, is considerable [24]. An example for such significant difference might be given when comparing the views on service interactions of automotive systems engineering and of service systems in aspects like functionality, parallelism, structure etc.

This paper considers service interaction problem in M2M environment using abstraction on device connectivity. Device connectivity management is context independent and common for all type of devices that acquire IoT connectivity using cellular or wireless technology. The presented algorithm for automatic discovery of service conflicts is based on standard reasoning. It may be used for implementation of M2M service orchestration and resolving service interactions by applying policies.

9. Conclusion

With the constant trend for increase of the number of smart devices equipped with sensors and actuators and used in different application areas, seamless service continuity is difficult to achieve due to the lack of dynamic provisioning.

Being smart, the devices need to connect in order to communicate with other devices. In this paper, a basic device connectivity model and a method for automatic service orchestration between M2M applications which add value to basic bearer selection procedure are proposed. It is important to mention that the proposed method for reasoning and resolving service interaction can be automated since the programmability of the reasoning algorithm. The formal description of the presented models can be translated from description logic into Ontology Web Language description where concepts are represented by classes, roles by properties and statements in the TBox and ABox as restrictions. There exist a number of ontology editors and frameworks for constructing domain models and knowledge-based applications with ontologies and reasoners to infer logical consequences from a knowledge base.

The proposed method for resolving service interaction using policies allows self-configuration of services. The level of human involvement in the network management can be reduced due to automatic orchestration of services provided by different service providers with coherence and consistency in order to provide adaptable service continuity to end users.

***Acknowledgements*:** The research is under the support of the project ДН07/10-2016 funded by National Science Fund, Ministry of Education and Science, Bulgaria.

R e f e r e n c e s

1. H u i, L., C. M i n. Research on the Distribution System Simulation of Large Company's Logistics under Internet of Things Based on Traveling Salesman Problem Solution. – Cybernetics and Information Technologies, Vol. **16**, 2016, No 5, pp.78-97.
2. W e y r i c h, M., C. E b e r t. Reference Architectures for the Internet of Things. – IEEE Software, January/February 2016, pp. 112-116. **https://www.computer.org/csdl/mags/so/ 2016/01/mso2016010112.pdf**
3. K i m, J., J. L e e, J. K i m, J. Y u n. M2M Service Platforms: Survey, Issues, and Enabling Technologies. – IEEE Communications Surveys & Tutorials, Vol. **16**, First Quarter 2014, No 1, pp. 61-76.
4. A p e l, S., C. K a s t n e r, B. G a r v i n. Exploring Feature Interactions in the Wild: The New Feature Interaction Challenge. – In: Proc. of 5th International Workshop on Feature-Oriented Software Development, FOSD, ACM, 2013, pp.1-8.
5. D a t t a, S., C. B o n n e t. A Lightweight Framework for Efficient M2M Device Management in OneM2M Architecture. – In: International Conference on Recent Advances in Internet of Things (RIoT'15), 2015, pp.1-6.
6. Open Mobile Alliance. Diagnostics and Monitoring Management Object. – OMA-TS-DiagMonTrapMO-V1_0-20090414-C, 2009.

7. Open Mobile Alliance. Diagnostics and Monitoring Trap Events Specifications. – OMA-TS-DiagMonTrapEvents-V1_2-20131008-A, 2013.
8. P e n c h e v a, E., I. A t a n a s o v. Detection of CAMEL Feature Interaction. – International Journal on Information Technology and Security, 2010, No 1, pp. 25-42.
9. A t a n a s o v, I., E. P e n c h e v a. CAMEL Service Interaction Detection. – International Journal on Information Technologies and Control, 2010, Issue 4, pp. 2-9.
10. A t a n a s o v, I., E. P e n c h e v a. A Formal Approach to Service Interaction Detection in Mobile Networks. – In: 10th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed Systems (SEPADS'11), Cambridge, UK, 2011, pp. 118-123
11. A t a n a s o v, I., E. P e n c h e v a. Reasoning on Service Interaction in Mobile networks. – International Journal of Computers and Communications, Vol. **15**, 2011, No 2, pp. 59-66.
12. 3GPP Technical Specification Group Services and System Aspects. – Policy and Charging Control Architecture, Release 13, V13.7.0, 2016.
13. A n d r o c e c, D., N. V r c e k. Ontologies for Platform as Service APIs Interoperability. – Cybernetics and Information Technologies, Vol. **16**, 2026, No 4, pp. 29-44.
14. C h e n, Y., Q. X i n, S. G u a n. Keynote Speech 1: Creating IoT Applications through Visual Programming and Service Orchestration. – In: 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference, Chongqing, China, 2016, pp. 1-7.
15. I m, C., C, J e o n g. ISOMP: An Instant Service-Orchestration Mobile M2M Platform. – Hindawi Mobile Information System, Vol. (2016), 2016 Article ID 7263729, pp.1-16.
16. G u p t a, H., S. B. N a t h, S. C h a k r a b o r t y, S. K. G h o s h. SDFog: A Software Defined Computing Architecture for QoS Aware Service Orchestration over Edge Devices. Cornell University Library, arXiv:1609.01190 [cs.NI].
17. L e e, N., H. L e e, W. R y u. Considerations for Web of Object Service Architecture on IoT Environment. – International Journal of Smart Home, Vol. **9**, 2015, No 1, pp. 195-202.
18. D'A n g e l o, G., S. F e r r e t t i, V. G h i n i. Simulation of the Internet of Things. Cornell University Library, arXiv:1605.04876 [cs.NI].
19. A p e l, S., J. A t l e e, L. B a r e s i, P. Z a v e. Feature Interactions: The next Generation. Report form Dagstuhl Seminar 14281, 2014, pp. 1-5.
20. M a t e r n a g h a n, C., K. T u r n e r. Policy Conflicts in Home Automation. – Computer Networks, Vol. **57**, 2013, Issue 12, pp. 2429-2241.
21. D o m i n g u e z, A. L. Detection of Feature Interactions in Automotive Active Safety Features. PhD Thesis, School of Computer Science, University of Waterloo, 2012.
22. L i n, Y. B., et al. EasyConnect: A Management System for IoT Devices and Its Applications for Interactive Design and Art. – IEEE Internet of Things, Vol. **2**, 2015, Issue 6, pp. 551-561.
23. P e d e r s e n, T., T. L e G u i l l y, A. P. R a v n, A. S k o u. A Method for Model Checking Feature Interactions. – In: 10th International Joint Conference on Software Technologies (ICSOFT'15), Colmar, Alsace, France, 2015, pp. 1-10.
24. Z a v e, P. Modularity in Distributed Feature Composition. – In: Software Requirements and Design: The Work of Michael Jackson. Good Friends Publishing, 2010, pp. 267-290.