RESTful Interfaces for Application Initiated D2D Communications

Evelina Pencheva
Faculty of Telecommunications
Technical University of Sofia
Sofia, Bulgaria

Ivaylo Atanasov
Faculty of Telecommunications
Technical University of Sofia
Sofia, Bulgaria

Denitsa Kireva-Mihova
Faculty of Telecommunications
Technical University of Sofia
Sofia, Bulgaria

Ventsislav Trifonov
Faculty of Telecommunications
Technical University of Sofia
Sofia, Bulgaria

Abstract— Device-to-Device (D2D) communications feature ultra low latency and efficient resource utilization. D2D communications may be empowered by Multi-access Edge Computing (MEC) which brings cloud capabilities for storage and processing into radio access network. In this paper, we present an approach to define RESTful interfaces for open access to proximity-based service, which may be used by mobile edge applications to initiate device proximity discovery and direct communications. The approach is illustrated by typical use cases, data model, interface definition and proximity-based service state models, which are formally described and verified.

Keywords—Multi-access Edge Computing, Traffic Offloading, Application Programming Interfaces, Finite State Machines

I. INTRODUCTION

The increasing demand for multimedia communications with high bandwidth and low latency requirements becomes a challenge for cellular networks. A new paradigm that may face this challenge is Device-to-Device (D2D) communications. D2D communications allow user equipments (UEs) in close proximity to communicate directly and thus enabling resilient radio resource management [1], [2]. In case of network congestion, D2D connectivity enables traffic offloading in dense environment. D2D communications may support different applications such as local data services for broadcasting and information sharing, data and computational offloading, where device with a good internet connectivity cashes the data and transmits it to other devices, coverage extension services, where a device may act as a relay between the base station and other devices, and Internet of Things applications enabling autonomous connectivity and communications between devices.

Considerable research is ongoing in the field of offloading techniques. In [3], the authors present a smart base station-assisted partial-flow device-to-device offloading system that provides seamless video streaming services to clients by effectively offloading parts of the video traffic between devices in order to alleviate the cellular network traffic load. The offloading utility of communications between devices may be maximized by proposing an optimal content pushing strategy based on the user interests and sharing willingness [4]. An offloading scheme of D2D communications on multi-radio technologies is proposed in [5]. The scheme can achieve high link spectrum efficiency without sacrificing the available terminal density. In [6], the authors formulate an optimization problem in D2D communications to maximize the number of users served and reduce the number of access points deployed while satisfying a set of system constraints. An incentive framework of D2D offloading, where the operator encourages some users acting as D2D transmitters to broadcast their popular contents to nearby region to improve operators' overall economic efficiency is proposed in [7]. The authors of [8] investigate the relationship between offloading gain of the system and energy cost of each device involved in the process. Caching schemes in mobile devices through deriving the minimum distance between two actively offloaded mobile user receivers to optimize the D2D throughput while guaranteeing the D2D channel quality are proposed in [9].

Multi-access Edge Computing (MEC) exposes great advantages in support of D2D communication technology. MEC distributes IT and cloud capabilities for storage and processing in the radio access network. The proximity to the end users in combination with D2D communications enables bandwidth hungry multimedia applications and may reduce traffic load in the radio access network. The utilization of radio resources may be improved with edge caching and computing on smart devices. Computation offloading and content sharing problems via D2D communications are discussed in [10].

D2D communications are based on Proximity-based Service which allows direct communication between User Equipments (UEs). As to [11], Proximity-based Service (ProSe) allows identification of ProSe-enabled UEs which are in proximity, using cellular or wireless technology, as well as establishment of communication paths between two or more ProSe-enabled UEs that are in direct communication range. The ProSe Direct Communication path could use Long Term Evolution (LTE) access network or Wireless Local Area

The research is within the frame of project DH07/10-2016, funded by National Science Fund, Ministry of Education and Science, Bulgaria

network (WLAN). An approach to open access to ProSe functionality based on network supported discovery is proposed in [12]. The approach is based on Service Oriented Architecture and exploits Web Services. In this paper, we define RESTfull interfaces for open access to ProSe which may be implemented in MEC environment. In current standards, it is the UE that initiates proximity discovery and communication offloading. With the proposed approach, it is a mobile edge application hosted by 3rd party or network operator that initiates the procedures and thus controls the ProSe-enabled UEs connectivity. The research motivation is to delegate the offloading control logic to authorized applications that can make a decision for offloading based on access network congestion, device location, quality of service, etc. D2D offloading scheme may be useful for users which are at cell edges, inside isolated environment like basements or large buildings and require better quality of service. The proposed RESTful interfaces provide open access to ProSe functionality in the network for 3rd party applications, which is a new source of revenue generation for network operators and an opportunity for service providers to create attractive and competitive applications [13].

The paper is structured as follows. Section II provides informative description of the proposed functionality illustrated by use cases. Section III describes the data model and actual interface definition. Section IV presents the proposed ProSe state models and their formal verification. The conclusion summarizes the author contribution.

II. SERVICE DESCRIPTION

The proposed Application-driven ProSe (ADPS) enables applications to identify that ProSe-enabled UEs are in proximity and to initiate direct D2D communications. The ADPS is enabled as RESTful interface and accepts four logical kinds of requests from mobile edge applications:

- request to manage event registrations,
- request to activate proximity discovery,
- request to manage proximity alert registrations, and
- request to notify UEs about their proximity.

Mobile edge applications can subscriber with the ADPS for notifications about registrations of ProSe enabled UE and applications. When ever such events occur, the service generates notifications to inform the mobile edge applications about the event. Having information about registered ProSe enabled UE associated with the mobile edge host, a mobile edge application may initiate proximity discovery of ProSe enabled UEs and it may subscribe to receive proximity alerts.

The mobile edge host needs to incorporate functionality of ProSe Function and ProSe Application Server. The ProSe Function is the logical function that is used for network related actions required for ProSe. For the purpose of Application-driven ProSe the required function is EPC-level ProSe Discovery [11]. The ProSe Application Server stores and maps user and application identities, and maintains permission information for restricted discovery [11].

The overall message flow for EPC-level ProSe Discovery consists of the following steps: UE registers with ProSe Function, the user register a specific application, proximity discovery, which includes location reporting of interested UE, and proximity alert.

To obtain ProSe service a ProSe-enabled UE needs to register with the ProSe Function. A mobile edge application subscribes to receive notifications about ProSe enabled UE in order to control proximity discovery procedure. When a ProSe enabled UE registers for ProSe, the application is notified. The user may activate D2D communications in the context of specific application. To activate ProSe features such as EPC-level ProSe Discovery for a specific application, the UE registers the application with the ProSe Function. So, the mobile edge application needs to subscribe for receiving notifications about application registration. When a user registers with a 3rd party application server, he/she is designated an Application Layer User ID (e.g. ALUID), and the mobile edge application is notified. The message flow for UE registration and application registration is shown in Fig.1.

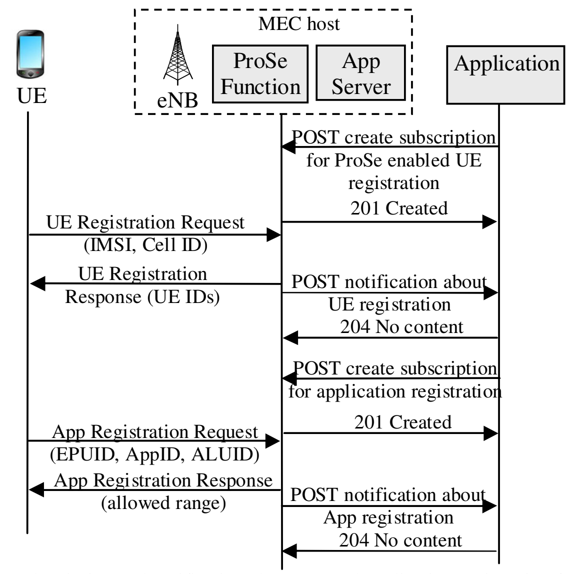

Fig.1 Subscription and notification about UE and application registration for ProSe

The mobile edge application initiates proximity discovery for registered ProSe enabled UEs currently associated with the mobile edge host (possibly indicating a window of time during which the request is valid). The application subscribes for notifications about UEs' proximity. In response, ProSe Function activates location updates for UE 1 and UE 2. These location updates can be periodic, based on a trigger, or a combination of both. The UEs' locations are reported to the ProSe Function intermittently. Whenever ProSe Function receives location updates for UE 1 and/or UE 2, it performs proximity analysis on UE 1 and UE 2's locations. When ProSe Function detects that the UEs are in proximity, it notifies the mobile edge application, which in turn decides whether to alert or not the UEs about their proximity. The message flow for proximity request, location reporting and proximity alert is shown in Fig.2.

The UE may decide to de-register for ProSe (e.g. when there are no ProSe-enabled applications activated on the UE) and in case of active subscription, the mobile edge application is notified. At any time the mobile edge application may initiate UE de-registration for ProSe (e.g. due to change in UE location, termination of corresponding application or due to completion of certain event).

The mobile edge application may decide to cancel Proximity Request it sent earlier (e.g. due to change in UE location, termination of corresponding application or due to completion of certain event).

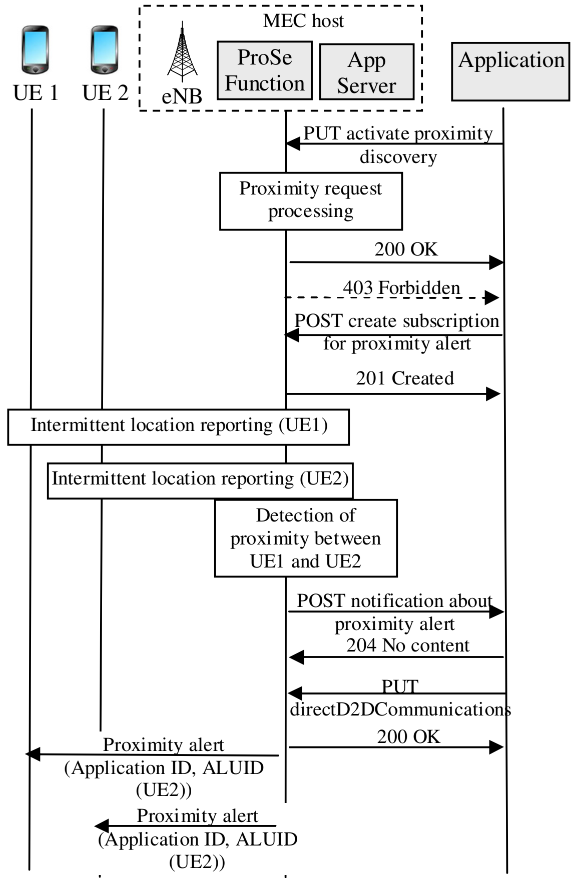

Fig.2 Proximity request, location reporting and proximity alert

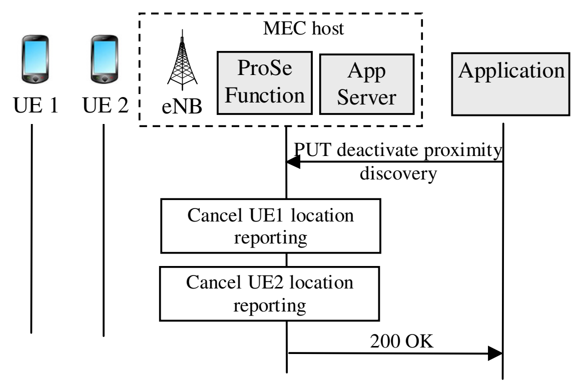

Fig.3 Application initiated UE de-registration

III. Data Model

This section defines data structures that are used in the resource representation.

The proSeEnabledUEs resource is a placeholder for one or more <proSeEnableUE> resources.

The <proSeEnableUE> resource represents information about ProSe enabled UE. The proSeUEApplications resource is a placeholder for one or more specific applications for the ProSe enabled UE. The deregistrationAction resource contains the action to be executed – at any time the mobile edge application may initiate UE de-registration. The ueLocationInfo resource contains information about UE location as defined in [14]. The ueProximities resource is a placeholder for one or more <ueProximity> resources, where the <ueProximity> resource represents information about proximity request. The proDiscoveryAction resource contains the proximity discovery action to be executed. The directD2DcomAction resource contains action related to notification of UEs about their proximity and thus allowing direct D2D communications.

Fig.4 shows the tree structure representing the resources.

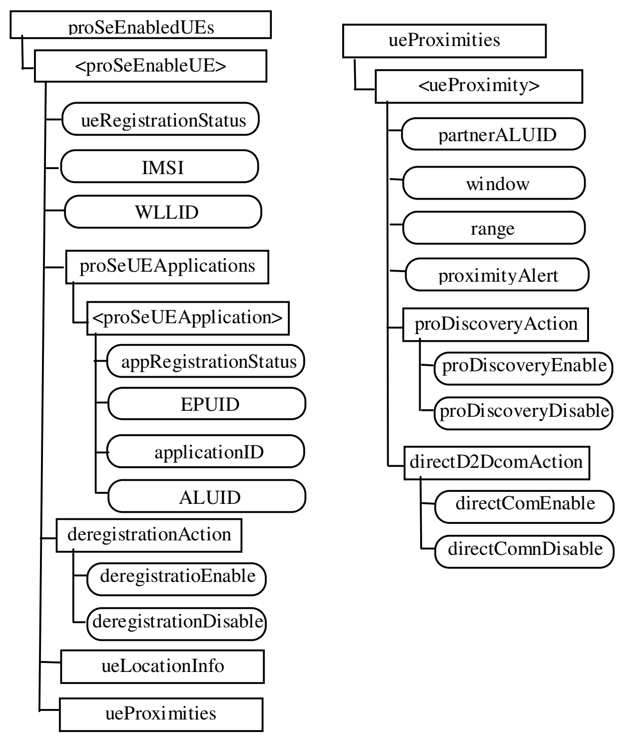

Fig.4 Resource structure

Table 1 describes the semantics of resource attributes, and Table 2 shows the resources and methods overview.

IV. Proximity-based Service Models

Models, representing the ProSe service status as seen by the mobile edge application, and by the network, are proposed in the section. These models have to expose equivalent behavior, i.e. they have to be synchronized. The synchronized behavior of the models allows proving in a mathematically formalized manner that the approach is consistently implementable. Mathematical formalism for equivalence of behavior is used to generate model-based test situations in order to demonstrate compliance of system's implementation with its specification.

TABLE I. RESOURCE ATTRIBUTES

Attribute name	Access	Description
ueRegistrationStatus	RO	Indicates the ProSe registration status
IMSI	RO	International Mobile Subscriber Identity
WLLID	RO	WLAN Link Layer ID
appRegistrationStatus	RO	Indicates the application registration status
EPUID	RO	EPC ProSe User ID for the authorized UE as defined in [11]
applicationID	RO	Indicates the application ID.
ALUID	RO	Application Layer User ID as defined in [11]
deregistrationEnable	RW	The action that allows to enable the ProSe deregistration.
deregistrationDisable	RW	The action that allows to disable the ProSe deregistration.
partnerALUID	RW	Indicates the ALUID of the target UE.
window	RW	Indicates the time period during which the request is valid.
range	RW	A requested range class for this application chosen from the set of allowed range classes.
proximityAlert	RO	Indicates whether the UEs are in proximity.
proDiscoveryEnable	RW	The action that allows to enable proximity discovery.
proDiscoveryDisable	RW	The action that allows to disable proximity discovery.
directComEnable	RW	The action that enables direct D2D communication.
directComDisable	RW	The action that disables direct D2D communication.

The mobile edge host needs to maintain synchronized views on the ProSe state as seen by the ProSe application and by the ProSe function. Fig.5 illustrates the simplified mobile edge application view on the ProSe state. The transitions in the state model are driven by methods of ADPS interfaces.

In Null state, the RAN is not congested and there is no need from ProSe Application point of view to initiate offloading. In Null state, the ProSe Application may be notified about ProSe-enabled UE and/or application registration, as well as for UE deregistration. In Null state, the ProSe Application may be notified about RAN congestion and it sends to the network a proximity request to initiate in the congested cell a discovery of ProSe-enabled UE which are in proximity each other. This results in transition to InitiateProximityDiscovery state. A transition to ReportProximity state occurs when the ProSe Application receives an acknowledgement of the proximity request from the network. In ReportProximity state, the ProSe Application may be notified about UEs proximity and the result is transition to Null state. In ReportProximity state, the ProSe Application may be notified about the end of proximity alerting or an error in proximity alerting.

TABLE II. RESOURCES AND METHODS

Resource name	Resource URI	HTTP methods
proSeEnabledUEs	/proSeEnabledUEs	GET
proSeEnabledUE	/proSeEnabledUEs/ proSeEnabledUE	POST, GET, DELETE
proSeUEApplications	/proSeEnabledUEs/ proSeEnabledUE/ proSeUEApplications	GET
proSeUEApplication	/proSeEnabledUEs/ proSeEnabledUE/ proSeUEApplications/ proSeUEApplication	POST, GET, DELETE
deregistrationAction	/proSeEnabledUEs/ proSeEnabledUE/ deregistrationAction	GET, PUT
ueLocationInfo	/proSeEnabledUEs/ proSeEnabledUE/ ueLocationInfo	POST, GET, DELETE
ueProximities	/proSeEnabledUEs/ proSeEnabledUE/ ueProximities	GET
ueProximity	/proSeEnabledUEs/ proSeEnabledUE/ ueProximities/ ueProximity	POST, GET, DELETE
proDiscoveryAction	/proSeEnabledUEs/ proSeEnabledUE/ ueProximities/ueProximity/ proDiscoveryAction	GET, PUT
directD2DcomAction	/proSeEnabledUEs/ proSeEnabledUE/ ueProximities/ ueProximity/ directD2DcomAction	GET, PUT

Fig.6 illustrates the simplified network view on the state of ProSe.

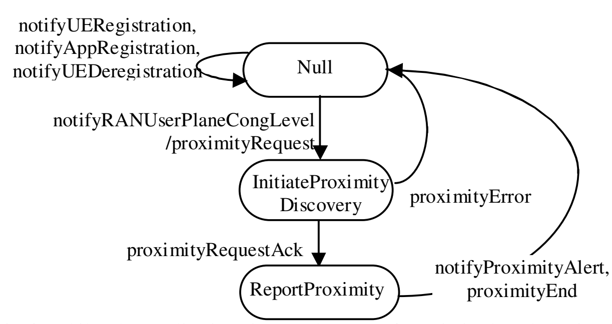

Fig.5 Mobile edge application view on the state of Proximity-based Service

In Registration state, the ProSe function receives registrations from ProSe-enabled UE and applications. The transition to ProximityRequest state occurs, when the ProSe

Application sends a request to report proximity. In IdentityMapping state, the ProSe function requests from the App Server to assign respective UE IDs for the ProSe users. In ProximityValidation state, the proximity request is validated. In LocationReporting state, UEs' locations are reported. The transition to ProximityAlerting state occurs when the ProSe function detects that the UEs are in proximity, and the ProSe function notifies the ProSe Application and the respective UEs.

A detailed description of network assisted ProSe discovery may be found in [11].

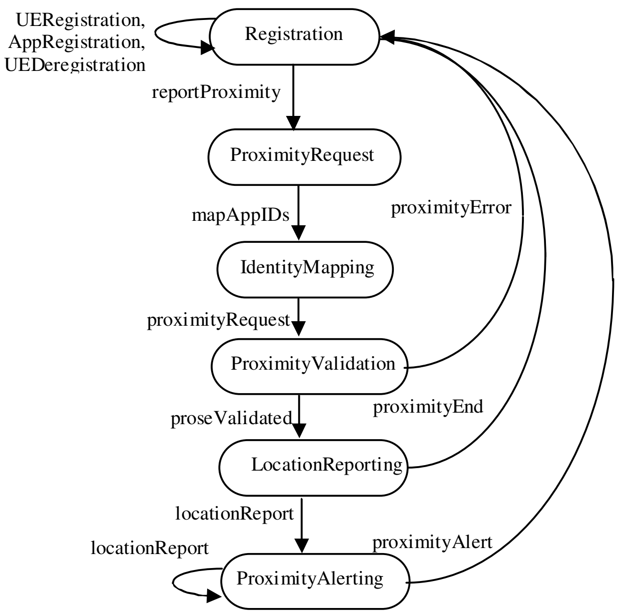

Fig.6 ProSe function view on the state of Proximity-based Service

Formal description of state machines is provided using the notation of Labeled Transition Systems (LTS).

Definition: A *Labeled Transition System* (LTS) is a quadruple $(S, Act, \rightarrow, s_0)$, where S is countable set of states, Act is a countable set of elementary actions, $\rightarrow \subseteq S \times Act \times S$ is a set of transitions, and $s_0 \in S$ is the set of initial states.

By $T_{App} = (S_{App}, Act_{App}, \rightarrow_{App}, s_0^{App})$ it is denoted an LTS representing the ProSe Application view on the ProSe state, where:

S_{App} = {Null [s_1^A], InitiateProximityDiscovery [s_2^A], ReportProximity [s_3^A]};

Act_{App} = { notifyUERegistration [t_1^A], notifyAppRegistration [t_2^A], notifyUEDeregistration [t_3^A], notifyRANUserPlaneCongLevel [t_4^A], proximityRequestAck [t_5^A], proximityError [t_6^A], proximityEnd [t_7^A], notifyProximityAlert [t_8^A]};

$\rightarrow_{App} = \{ (s_1^A\ t_1^A\ s_1^A), (s_1^A\ t_2^A\ s_1^A), (s_1^A\ t_3^A\ s_1^A), (s_1^A\ t_4^A\ s_2^A), (s_2^A\ t_5^A\ s_3^A), (s_2^A\ t_6^A\ s_1^A), (s_3^A\ t_7^A\ s_1^A), (s_3^A\ t_8^A\ s_1^A) \}$;

s_0^{App} = *{Null}*.

Short notations of state and transition names are given in brackets.

By $T_N = (S_N, Act_N, \rightarrow_N, s_0^N)$ it is denoted an LTS representing the ProSe function view on the ProSe state, where:

S_N= {Registration [s_1^N], ProximityRequest [s_2^N], IdentityMapping [s_3^N], ProximityValidation [s_4^N], LocationReporting [s_5^N], ProximityAlerting [s_6^N]};

Act_N = { UERegistration [t_1^N], AppRegistration [t_2^N], UEDeregistration [t_3^N], reportProximity[t_4^N], mapAppIDs [t_5^N], proximityRequest [t_6^N], proseValidated[t_7^N], locationReport[t_8^N], proximityAlert [t_9^N], proximityEnd [t_{10}^N], proximityError [t_{11}^N] };

$\rightarrow_N = \{ (s_1^N\ t_1^N\ s_1^N), (s_1^N\ t_2^N\ s_1^N), (s_1^N\ t_3^N\ s_1^N), (s_1^N\ t_4^N\ s_2^N), (s_2^N\ t_5^N\ s_3^N), (s_3^N\ t_6^N\ s_4^N), (s_4^N\ t_7^N\ s_5^N), (s_4^N\ t_{11}^N\ s_1^N), (s_5^N\ t_8^N\ s_6^N), (s_6^N\ t_8^N\ s_6^N), (s_5^N\ t_{10}^N\ s_1^N), (s_6^N\ t_9^N\ s_1^N) \}$;

- s_0^N = *{Registration}*.

Having formal description of the models representing ProSe status as seen by ProSe Application and ProSe function, we can prove that these models are synchronized i.e. they expose equivalent behavior.

Intuitively, in terms of observed behavior, two LTSs are equivalent if one LTS displays a final result and the other LTS displays the same result. The idea of equivalence is formalized by the concept of bisimilarity [15]. In practice, strong bisimilarity puts strong conditions for equivalence which are not always necessary. Weak bisimilarity allows internal transitions to be ignored.

Proposition: T_{App} and T_N are weakly bisimilar.

Proof: To prove the bisimilarity between two LTSs, it has to be proved that there exists a bisimilar relation between their states. We identify the following relation between the states of of T_{App} and T_N: Null and Registration, InitiateProximity-Discovery and ProximityRequest, ReportProximity and LocationReporting. This relation is denoted by U_{AppN} where $U_{AppN} = \{(s_1^A, s_1^N), (s_2^A, s_2^N), (s_3^A, s_5^N)\}$. Then:

1. The ProSe Application is notified when a ProSe-enabled UE and/or application registers with the ProSe

function: for ($s_1^A\ t_1^A\ s_1^A$), ($s_1^A\ t_2^A\ s_1^A$) $\exists$ ($s_1^N\ t_1^N\ s_1^N$), ($s_1^N\ t_2^N\ s_1^N$).

2. The ProSe Application is notified when a ProSe-enabled UE is deregistered: for ($s_1^A\ t_3^A\ s_1^A$) $\exists$ ($s_1^N\ t_3^N\ s_1^N$).

3. When the ProSe Application is notified about network congestion it sends a proximity request to the network: for ($s_1^A\ t_4^A\ s_2^A$) $\exists$ ($s_1^N\ t_4^N\ s_2^N$).

4. The proximity request is validated: for ($s_2^A\ t_5^A\ s_3^A$) $\exists$ ($s_2^N\ t_5^N\ s_3^N$), ($s_3^N\ t_6^N\ s_4^N$), ($s_4^N\ t_7^N\ s_5^N$).

5. The proximity request is not validated: for ($s_2^A\ t_6^A\ s_1^A$) $\exists$ ($s_2^N\ t_5^N\ s_3^N$), ($s_3^N\ t_6^N\ s_4^N$), ($s_4^N\ t_{11}^N\ s_1^N$).

6. The ProSe Application is notified when the ProSe function detects that UEs are in proximity: for ($s_3^A\ t_8^A\ s_1^A$) $\exists$ ($s_5^N\ t_8^N\ s_6^N$), ($s_6^N\ t_8^N\ s_6^N$), ($s_6^N\ t_9^N\ s_1^N$).

7. The ProSe Application is notified that the proximity reporting is ended: for ($s_3^A\ t_7^A\ s_1^A$) $\exists$ ($s_5^N\ t_{10}^N\ s_1^N$).

Therefore T_{App} and T_{N} are weakly bisimilar.■

The synchronized behaviour of the models allows proving in a mathematically formalized manner that the approach is consistently implementable. Mathematical formalism for equivalence of behaviour is used to generate model-based test situations in order to demonstrate compliance of a system's implementation with its specification.

V. Conclusion

In this paper, we propose RESTful interfaces that provide open access to Proximity-based service in radio access network. The proposed mobile edge service allows 3rd party applications to initiate UE's proximity discovery in order to trigger D2D communications and traffic offloading. The research novelty is in delegating the control of ProSe functions to 3rd party applications. With existing standardized solutions it is the UE that decides when to switch off to D2D communications. Delegating the control to dedicated applications allows a service provider or network operator to define a policy for traffic offloading, e.g. based on radio access network congestion level, device location or quality of service.

The open access to ProSe functionality might be seen as an approach that is three-fold beneficial. The approach gain for operators is consisted mainly by its inherent capability to mitigate the congestion-related situations by additional offload procedures. The applications i.e. the third party might be involved in the very same process while pursuing the fulfillment of own quality of service promises given, especially regarding losses and latency. Finally, the indirect aspect of improvement is the one about the end-user quality of experience which is going to be affected inevitably. All this has a price of procedures complexity increment within the operator's network but it seems that gain outweighs it.

References

[1] P. Gandotra, R. K. Jha, S. Jain "A survey on device-to-device (D2D) communication: Architecture and security issues," Journal of Network and Computer Applications, Elsevier, vol.78, January, 2017, pp.9-29.

[2] U. Narayana, D. K. Sanyal, An overview of device-to-device communication in cellular networks, ICT Express, October 2017, pp.1-6.

[3] G. S. Park, W. Kim, S. H. Jeong, H. Song. "Smart Base Station-Assisted Partial-Flow Device-to-Device Offloading System for Video Streaming Services," IEEE Transactions on Mobile Computing, vol. 16, no. 9, pp. 2639-2655, Sept., 2017.

[4] Y. Pan, C. Pan, H. Zhu, Q. Z. Ahmed, M. Chen J. Wang. "Content offloading via D2D communications based on user interests and sharing willingness," 2017 IEEE International Conference on Communications (ICC), Paris, France, 2017, pp. 1-6.

[5] C. Liu, C. He, W. Meng. "A tractable Multi-RATs offloading Scheme on D2D Communications," IEEE Access, September, vol 5, 2017, pp.20841-20851.

[6] N. Abbas, Z. Dawy, H. Hajj, S. Sharafeddine F. Filali. "Traffic offloading with maximum user capacity in dense D2D cooperative networks," IEEE International Conference on Communications (ICC), Paris, 2017, pp. 1-6.

[7] B. Shang, L. Zhao, K. C. Chen. "Operator's Economy of Device-to-Device Offloading in Underlaying Cellular Networks," IEEE Communications Letters, vol. 21, no. 4, pp. 865-868, April 2017.

[8] B. Chen, C. Yang, A. F. Molisch, "Cache-Enabled Device-to-Device Communications: Offloading Gain and Energy Cost," IEEE Transactions on Wireless Communications, vol. 16, no. 7, pp. 4519-4536, July 2017.

[9] X. Zhang, Q. Zhu. "Distributed mobile devices caching over edge computing wireless networks," 2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Atlanta, GA, 2017, pp. 127-132.

[10] J. Cao, A. Castiglione, G. Motta, F. Pop, Y. Yang and W. Zhou, "Human-Driven Edge Computing and Communication: Part 1," IEEE Communications Magazine, vol. 55, no. 11, pp. 70-71, November 2017.

[11] 3GPP TS 23.303 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Proximity-based services (ProSe); Stage 2, (Release 14), v14.1.0, 2017.

[12] Pencheva, E., D. Kireva, I. Atansov, V. Trifonov. "An Approach to Provide Open Access to Proximity-Based Service", International Journal on Information Technologies and Security, No1, vol.10, 2018, pp.13-24.

[13] Pencheva, E., I. Atanasov. "Third Party Application Control on Quality of Service in IP based Multimedia Networks", International Journal "Information Systems Frontiers", dedicated issue on "Ubiquitous Multimedia Services", 2012, vol.14, pp.555-569.

[14] ETSI GS MEC 013 Mobile Edge Computing (MEC); Location API, v1.1.1, 2017

[15] G. Pola, C. Manes, A. van der Schaft and M. D. Di Benedetto, "Bisimulation Equivalence of Discrete-Time Stochastic Linear Control Systems," IEEE Transactions on Automatic Control, vol. PP, no. 99, 2017, pp. 1-1.