IEEE Consumer Electronics Magazine, October 2016, pp.77-83, DOI: 10.1109/MCE.2016.2590100

MEC come home! Connecting things in future smart homes using LTE D2D communications

Carlo Vallati, Antonio Virdis, Enzo Mingozzi, Giovanni Stea
Dipartimento di Ingegneria dell'Informazione, University of Pisa
Largo Lucio Lazzarino 1, I-56122, Pisa, Italy
{c.vallati, a.virdis, e.mingozzi, g.stea}@iet.unipi.it

***Abstract*—Future 5G cellular networks are expected to play a major role in supporting the future Internet of Things (IoT), due to their ubiquitous coverage, plug-and-play configuration and embedded security. Besides connectivity, however, IoT will need computation and storage in proximity of sensors and actuators to support time-critical and opportunistic applications. To this aim, the introduction of Mobile Edge Computing (MEC) is currently under standardization as a novel paradigm to enable hosting of applications straight into the network. In this work we analyze solutions to bring MEC functionalities as close as possible to end users and smart objects. First, a smart-home system is designed as a reference small-scale IoT system to derive network requirements; then alternative network configurations to support such requirements are analyzed to highlight their pros and cons. In particular, we show how LTE Device-to-Device (D2D) operation mode can be exploited to guarantee proximity communication with reduced costs. Finally, the expected benefits for operators are assessed via simulation.**

Keywords—MEC; CoAP; IoT; LTE; LTE-Advanced; D2D; Smart Home

I. INTRODUCTION

Mobile Edge Computing (MEC) is currently under standardization as a novel paradigm expected to enrich future broadband communication networks [1],[2]. With MEC, traditional networks will be empowered by placing cloud-computing-like capabilities within the Radio Access Network (RAN), in a MEC server located in close proximity to end users. Such distributed computing and storage infrastructure will enable the deployment of applications and services at the edge of the network, allowing both operators to offer a virtualized environment to enterprise customers, and industries to implement applications and services close to end users.

MEC is currently recognized as a key enabling technology for a wide range of scenarios and use-cases. Among them, however, the future Internet of Things (IoT) is the one expected to reap the highest benefit from a run-time environment distributed through the network. Current IoT systems are composed of smart objects – sensors and actuators – which are controlled by IoT applications running in a cloud environment deployed in remote data centers. Mission-critical applications such as closed-loop control logic or opportunistic applications based on proximity will require direct Machine-to-Machine (M2M) interactions with *low latency* and *preservation of locality*, both of which are unachievable if applications run in a cloud environment. To unleash the full potential of future IoT, the integration of local runtime environment such as MEC is envisaged. MEC fits into a new computing paradigm, the Fog computing [4] currently under standardization by the *OpenFog consortium* [17]. Fog computing extends elements of computation, networking and storage across the cloud through to the edge of the network, closer to end users and devices. In this context, MEC can be considered as a particular Fog implementation where computing and storage are managed and offered by network operators through their communication networks.

In the IoT context, 4G/5G cellular networks, and specifically LTE, will play a major role [3]. Although different communication technologies for IoT are available, e.g. IEEE 802.15.4 or Bluetooth low-power, LTE provides unique advantages considering its widespread infrastructures already deployed on a large scale, ubiquitous coverage and the service reliability offered by a mature wireless standard deployed on licensed spectrum [5]. In addition, LTE offers by design several features that are mandatory to support M2M communications, such as auto-configuration for plug-and-play devices that do not require human intervention, security through data encryption and authentication, Quality of Service (QoS) support for data delivery with stringent real-time requirements, and energy-saving mechanisms for battery-powered devices [6]. Besides the traditional Device-to-Infrastructure (D2I) communication, the LTE standard includes a Device-to-Device (D2D) mode that supports direct interaction between devices without relaying the communication through the base-station node. Such operation mode has been recently introduced to further support M2M applications, facilitating the discovery of neighboring devices and direct communication with reduced costs. Although the LTE standard offers several technical solutions amenable to M2M interactions, it does not suggest one practical solution specifically [7], and several configuration options are at hands to support communication between a MEC server and physically proximate smart objects.

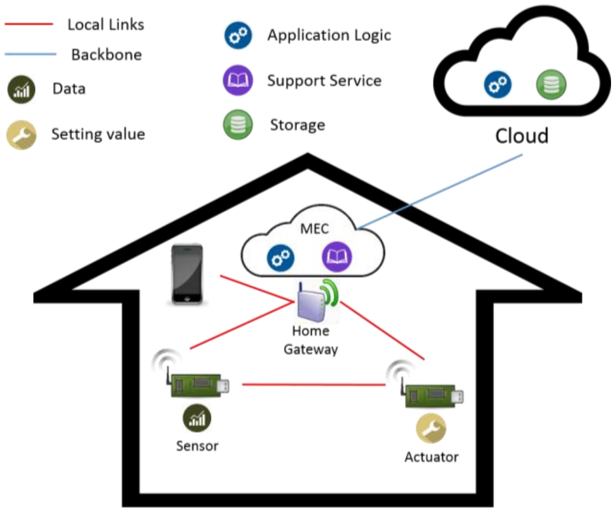

Figure 1 - Smart home IoT system architecture.

In this work, which extends the one appeared as [16], we analyze how to deploy MEC closer to smart objects to realize an IoT Gateway with truly *direct* M2M interactions in future LTE networks. MEC specifications do not mandate specific deployment options [2]. However, a MEC server will be installed by definition at the edge of the network: in an LTE deployment, this means at best co-located with eNodeBs (eNBs). However, such an architecture will still have to cover too large an area, e.g. a macro cell in the order of square kilometers. Our goal is to analyze possible solutions to enable MEC logic to be moved much closer, i.e. directly next to sensors. Our reference use-case is a smart home, which is representative of small-scale IoT systems characterized by limited spatial occupancy and localized communications. In this context, a deployment of a small-scale IoT system based on the Constrained Application Protocol (CoAP) is designed, and its requirements on the network are derived. Then, we analyze different architectural options for network operators to offer MEC functionalities in this scenario. Among them, we propose a deployment that exploits D2D communications between the MEC server and smart objects to enable efficient direct communication between M2M applications and machines. D2D can ensure low-latency communication - a key requirement highlighted also in Fog architectures - by avoiding traffic rerouting through the network infrastructure. We compare this solution with one based on the legacy LTE configuration and another based on femtocells, and show how our D2D approach guarantees low latency, preserves locality, and offers plug-and-play auto-configuration. Finally, the expected benefits of each solution are assessed by means of simulations. In particular, we show D2D interactions can be exploited to utilize global network resources efficiently, which could be highly beneficial for service providers to achieve better utilization even on a large scale, and to improve end-user experience.

II. SMART-HOME IOT ARCHITECTURE

Smart-home systems have now reached commercial maturity with the appearing of smart appliances on the market. Their evolution is expected to create small house-wide ecosystems in which appliances from different vendors and applications created by different developers can interoperate seamlessly. Although availability of connected appliances using standard interfaces and protocols is already in production, supporting local applications that can run in proximity of smart objects is still a work in progress. In the past, most research efforts towards enabling a runtime environment for IoT applications have been focused on a cloud-based approach. Many commercial products and platforms adopted today in IoT solutions leverage the cloud computing infrastructure to guarantee easy and rapid deployment at a low cost. Recently, new research has designed a distributed runtime environment to execute applications in proximity of smart objects, e.g. the BETaaS EU project [8]. Although no standardized approach has been defined yet, recent progresses in virtualization techniques for mobile and constrained devices [10] are paving the way to enabling MEC-server functionalities on existing edge devices, e.g. home routers, set-top boxes, and smartphones.

Figure 1 shows the architecture of a smart-home system. Smart objects, e.g. appliances, are available as sensors and actuators to measure data or control systems, respectively. These devices connect to the home gateway, which is installed and managed by the network service operator. The connectivity provided by the gateway allows IoT applications running in the cloud to interact with smart objects, to implement IoT applications that have loose delay requirements or demand large storage, e.g. remote telemetry or historical data collection. Besides connectivity, the home gateway, if properly empowered, can also implement MEC-server functionalities, allowing local execution of IoT applications. This node, henceforth called *MEC node* for short, offers a local runtime environment to enable IoT applications with stringent delay requirements, or opportunistic applications exploiting proximity interactions, both of which cannot run in the cloud. Finally, these applications can expose an interface to human operators, e.g. for monitoring or configuration, through smartphones or web applications. Availability of computing and storage could enable the implementation of services to support smart objects operations as well: for example, a local directory service could be implemented to allow smart objects to lookup sensors or actuators available in proximity with certain capabilities or offering certain features.

The above architecture is general and can be mapped to many different IoT technologies or standards. Among the ongoing standardization efforts, however, the Constrained Application Protocol (CoAP) [9], specified by the CoRE Working Group of the Internet Engineering Task Force (IETF), is getting attention and is expected to become the standard for communication between smart objects and applications. CoAP follows the REpresentational State Transfer (REST) paradigm, the same one adopted by the HTTP protocol: smart objects offer

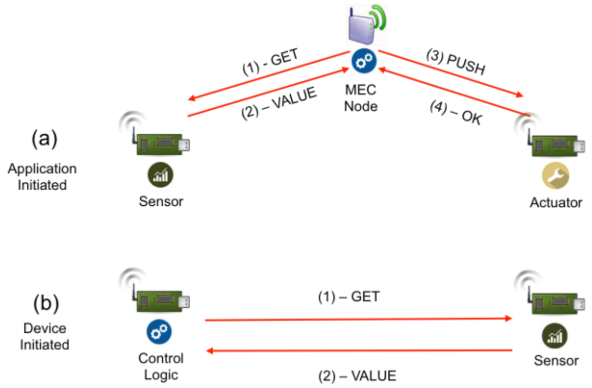

Figure 3 - M2M Interactions

their functionalities – retrieving a measurement or configuring a value – as *server resources*, accessed by applications or other smart objects as *clients*. Request/response transactions implement one-to-one interactions that are issued to perform regular operations. Four methods are defined: GET, PUSH, PUT and DELETE, which can be exploited by applications or smart objects to retrieve measurements from sensors or control actuators. With reference to the smart-home architecture presented in Figure 1, two possible M2M interactions are those illustrated in Figure 3, i.e., *application-initiated* and *device-initiated* interactions:

1. *Application-initiated interactions*, Figure 3(a), are initiated by the IoT application running on the MEC node that interacts with one or more smart objects. Consider as an example a closed-loop alarm application that controls an acoustic alarm (an actuator) based on the value gathered from a presence sensor: first, a measurement is retrieved from the sensor, then, if intrusion is detected, the alarm is triggered by sending a command to the actuator. In this case, the application behaves as a CoAP client, whereas smart objects expose their functionalities as CoAP servers. In particular, the sensor responds to a GET request by returning a measurement, while the actuator responds to a PUSH request by triggering its action.

2. *Device-initiated interactions*, Figure 3(b), are initiated by smart objects that communicate directly with each other. Optionally, a supporting service running on the MEC node – e.g., a directory service – might be contacted first to retrieve the identifier of the actual device to communicate with. Let us consider as an example an appliance that wants to avoid peaks of energy consumption. The smart object could interact with the smart meter to retrieve the current energy consumption and postpone energy-consuming operations to off-peak hours. In this case, the simple control logic implemented in the smart appliances behaves as a CoAP client issuing a GET request to a resource exposed by the smart meter.

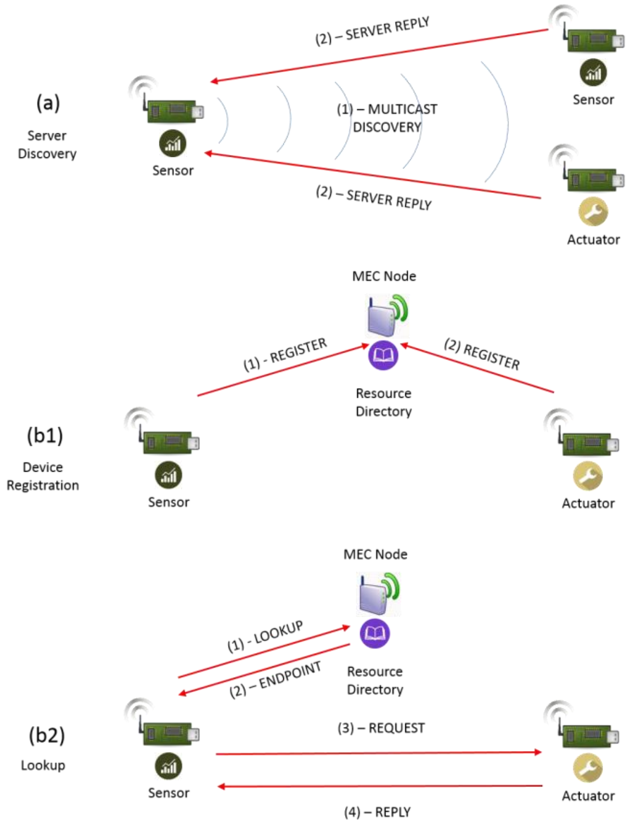

Figure 2 - Discovery Interactions

As highlighted in the last example, discovery is of paramount importance to enable auto-configuration. To this aim, CoAP defines a two-phase distributed discovery procedure. During the first phase, the client executes a *server discovery* to discover all the CoAP servers available within the same network. As illustrated in Figure 2(a), the client sends a multicast message to a special multicast group, and a CoAP server that receives the message replies and advertises its presence. Then, to obtain all the resources exposed by each server, the client executes a *resource discovery*. This is done by sending a GET request to a special resource by which the client retrieves the list of all available resources along with the description of the offered functionalities.

Direct resource discovery is infeasible in many scenarios, due to sleeping nodes or network limitations. Thus, an extension to the CoAP standard is currently under definition to include the option of implementing a directory service, which can be contacted by clients to look up for resources within the network. The extension, in particular, defines an entity named Resource Directory [11], which hosts information on resources exposed by CoAP servers available in a certain domain and exposes a look-up service for those resources to CoAP clients. With respect to the smart-home architecture, a Resource Directory instance can be installed on the local MEC node to support smart objects for their device-initiated interactions or applications. Resource Directory operations are depicted in Figure 2

IEEE Consumer Electronics Magazine, October 2016, pp.77-83, DOI: 10.1109/MCE.2016.2590100

(b1) and (b2). At bootstrap, devices register their presence to the local Resource Directory service, specifying the resources exposed and context information, e.g., their location, Figure 2(b1). The address of the Resource Directory could be well known a priori or discovered through network configuration. When a device needs the address of another device exposing a certain resource, e.g. the temperature in a given room, it queries the Resource Directory for it. After this look-up phase, the actual request is issued, Figure 2(b2).

Although MEC functionalities are usually offered inside the network to cover a large area, e.g. a neighborhood or a large building, the smart-home architecture presented here requires the MEC logic to be installed directly within the house. This is important for two reasons: first, to truly guarantee direct interaction with smart objects to achieve low latency; secondly, to preserve the domain represented by the house, guaranteeing protection of the locality information associated with each smart object. In addition, a MEC runtime environment localized within the house domain can support execution of external applications for example enabling an app marketplace [12].

Bringing MEC logic closer to smart objects is a challenge for the network configuration and for the network operators. In the following, we analyze possible options for achieving this goal, highlighting their pros and cons and respective expected benefits for the network operators.

III. NETWORK ARCHITECTURE

In this section we describe possible architectural solutions for implementing MEC-enabled smart-home systems within an LTE network. We assume that IoT devices will be using LTE as a communication technology. This allows a network operator to double as IoT service provider or to host third-party IoT applications with little extra costs. Moreover, LTE already offers built-in features, such as security, plug-and-play, QoS and reliability, which are clear requirements of commercial deployments.

From the point of view of the network a single smart-home environment is composed of tens of nodes interacting together with low bandwidth: it is thus a small-scale system. In such system, as described in Section II, all the interactions among nodes will be either between a MEC node and an IoT device, or between two IoT devices. In future scenarios, however, the coexistence of many such environments in the same geographic area (e.g. a large neighborhood with hundreds of houses) will raise issues that are typical of large-scale network systems, thus requiring non-trivial management on the network-operator side. In this context, network scalability can be achieved by working at two levels: first, tailoring the MEC architecture to the specific case of IoT systems, secondly implementing interactions among all involved nodes in an efficient manner.

The first level can be realized by placing the MEC node at the most appropriate location. According to the current vision, MEC functionalities are expected to be placed at the edge of the network, logically next to the eNB, as illustrated in Figure 4(a), and offering highly virtualized computation and storage resources installed and operated by service providers. Such features will be offered to users and third parties to deploy additional services and applications exploiting last-mile connection between UEs and eNBs. In this deployment one MEC server will end up covering a very large area, thus with high resource contention among possibly very far users. Bringing MEC servers closer to UEs, possibly inside the house and directly next to IoT devices, is thus a necessity, which comes with several challenges. Recent advancements in virtualization, enabled by hardware support, are making feasible empowering constrained devices, as home-network gateways (e.g. femto-cells), set-top boxes or even UEs, to offer a virtualized runtime environment [10]. Such solutions will open the way for two novel deployments, respectively placing a MEC server into a femto-cell or a UE, as shown in Figure 4 (b) and (c). This approach is compliant with the ETSI general architecture and can seamlessly be integrated into existing networks. It is worth mentioning that the deployment of the proposed solution is in accord with the

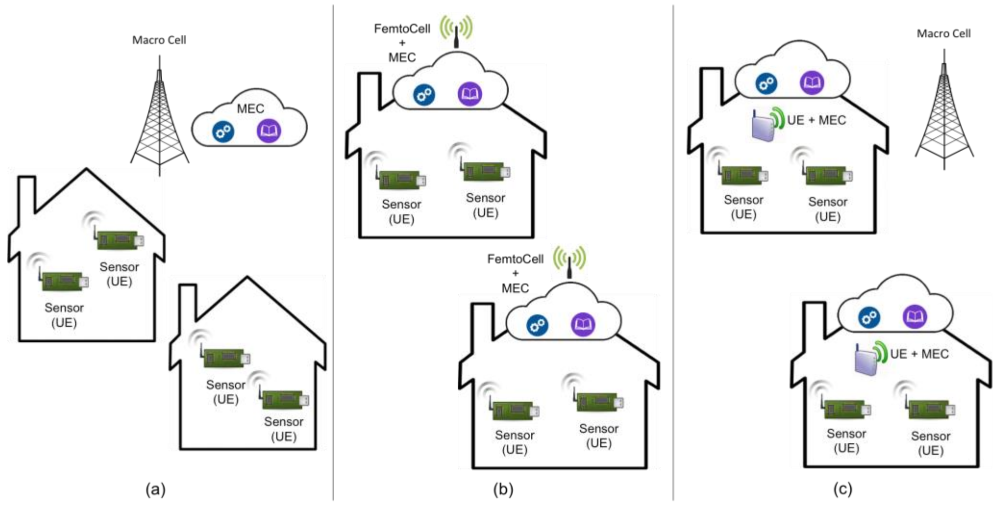

Figure 4 - MEC deployment alternatives

IEEE Consumer Electronics Magazine, October 2016, pp.77-83, DOI: 10.1109/MCE.2016.2590100

Table 1 - Comparison of MEC-enabled smart-home deployment options within an LTE network

	MEC-node 2 IoT-Device		IoT-Device 2 IoT-Device	
	D2I	**D2D**	**D2I**	**D2D**
MEC at Macro	Higher latency	Infeasible	High latency	Efficient communication Lower latency
MEC at Femto	Efficient communication High interference	Infeasible	Efficient communication High interference	Need for coordination
MEC at UE	Higher latency High interference	Lower latency Proximity Discovery	High interference	Efficient communication Lower latency

commercial practices already being adopted by wireless service operators: these often install network equipment, such as home routers or hotspot devices, inside the customer premises, i.e. in proximity of smart objects (e.g. appliances).

Bringing MEC logic closer to IoT devices produces benefits for both the operator and the end user. By ensuring local communications in fact, interactions take place within the smart home, enabling operations such as proximity discovery and reducing communication latency, thus enabling novel applications to be offered to the end user. From the network operator standpoint, proximity enables efficient MEC-node to IoT-device communications, both in terms of lower transmission power and reduced utilization of resources. On the other hand, these new deployments will result in ultra-dense scenarios, possibly characterized by a high interference. Although approaches for dynamic coordination are widely studied, femtocells are usually connected to the core network using commodity interfaces, such as Digital Subscriber Lines (xDSL), whose latency and limited bandwidth make fast coordination difficult, at best. In the next section we will assess the performance of such deployments from both the operator and end-user standpoints.

At the second level, D2D communications can be leveraged to go beyond D2I communication, fully exploiting the proximity of both MEC node and IoT devices. In LTE, D2D communications are managed by the eNB, which still signals grants as it does for D2I ones, but the data communication does not involve the eNB, and takes place directly from a sender to a receiver. The message exchange for a D2D communication to occur between two UEs is shown in Figure 5: UE1 signals to the eNB that it has new traffic, using the Random Access Channel request; then the eNB gives UE1 a grant large enough to transmit its Backlog Status Report, which indicates how large a backlog sits in UE1's queue. This way, the eNB can size the data grant to be sent to UE1 to transmit its traffic. At the same time, it will signal to UE2 to listen to the same resources granted to UE1, thus – if UE2 can physically hear UE1, direct communication will take place. Note that the signaling in Figure 5 is identical to that of a standard uplink communication, the only difference being that a) UE1 specifies UE2 instead of the eNB as a target, and b) the eNB also instructs UE2 to listen to the same resources. D2D communications can be either *unicast* or *multicast*. From the sender standpoint, the only difference is whether the intended target is a single UE address or a group ID address at the MAC layer. Multicast D2D transmissions will then reach all the UEs in hearing range of the sender. Multicast D2D transmissions have already been standardized by 3GPP, whereas unicast transmissions so far have only received attention from the scientific community [13],[14].

D2D communications would allow a MEC node to be implemented as an LTE UE, something which further extends the current MEC concept. In this case, all the local interactions of the IoT system will be implemented through D2D communications, both unicast and multicast. Specifically, IoT devices can exploit unicast D2D transmissions for device-initiated interactions and broadcast/multicast D2D ones for local discovery operations. Moreover, application-initiated interactions and interactions between devices and the MEC node for lookup and registration can be implemented through D2D unicast communication as well. Although the classic D2I communication paradigm is mainly adopted with femto-cells, D2D can be employed also if the MEC node is implemented as a femto-cell to improve the efficiency of at least some operations. Specifically, in this case, device-initiated interactions can be implemented as D2D communications. However, this would require additional spectrum coordination functionalities to be implemented into the femto-cell, so as to grant non-overlapping D2D communications inside the same femto-cell.

In Table 1 the pros and cons of the three deployments are presented. For each solution the advantages of each communication mode, i.e. D2I and D2D (when feasible), are highlighted with respect to the two communication patterns of the IoT system, MEC node to IoT device, and IoT device to IoT device, respectively. The table summarizes the observations presented so far. A subset of these remarks is verified by means of simulations in Section IV. As can be seen, when D2D is employed, a more efficient communication can be achieved, as lower latency and a more efficient spectrum management can be implemented. In the next section, such expected benefits are assessed by means of simulations.

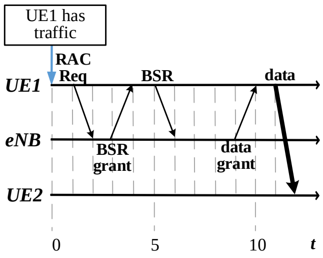

Figure 5 – D2D communication in LTE

e

tes

, 10 }

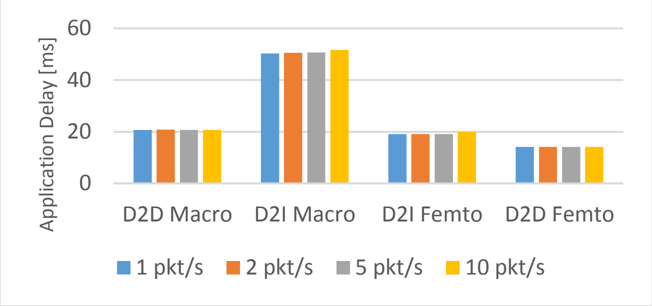

Figure 7 - Application delay

IV. Expected Benefits

In this section we compare the performance of D2D and D2I communications in an IoT system with multiple smart homes. We consider two main scenarios: a first one where a macro eNB serves all the smart homes and a second one where each of them is served by a femto eNB. This way we obtain a total of four simulation scenarios. IoT devices are randomly deployed within each smart home and communications occur only among IoT devices of the same smart home. D2D communication are scheduled on uplink resources, as these are expected to be less loaded and subject to less interference. Simulations are performed using SimuLTE [15], an OMNeT++-based system-level simulator of LTE-A networks. A summary of the simulation parameters is given in Table 2.

In Figure 6 we show that the application delay for the above configurations is higher in the case of D2I communications with the macro node, since every communication between nodes has to be relayed by the macro, which can even be further away from them. Performance can be improved by using either D2D or femto-eNBs. In the first case nodes just have to request resources to the macro-eNB, then communication can happen in a direct manner, whereas in the second one, communications are relayed by a node in proximity, which has better channel performance and ensures more efficient transmissions, thus significantly abating the delay. The best performance is achieved by combining the two above approaches, i.e. using D2D within a house served by a femto-eNB. In this last case in fact, resources are still requested to a serving node in proximity, and are then granted using D2D.

In Figure 8 and Figure 7 we evaluate the benefits for the operator by showing the percentage of used uplink (UL) and downlink (DL) transmission resources, considering the macro in the first two scenarios, and the sum of the femto-eNbs in the others. As we can see, UL resources do depend on the traffic volume, and are lower when using D2D with the macro and D2I with the femto. In the first case most of the communications occurring between couples of endpoints located in different homes can benefit from frequency reuse, whereas in the second UEs experience better channel conditions, thus can achieve the same data rate using fewer resources. Finally, DL resource utilization is shown only for D2I communications, as D2D one occurs only in UL. As we can see, using either macro- or femto-eNBs lead to similar results. In fact, the stronger transmission power used by the macro node compensates for its higher distance to the homes, hence to devices, which is far shorter in the case of femto-eNBs. However, operators may select one solution or the other also based on considerations on power consumption, i.e. depending on which node has better energy efficiency and on the fact that the energy bill for femto-eNBs can be expected to be paid by the end user.

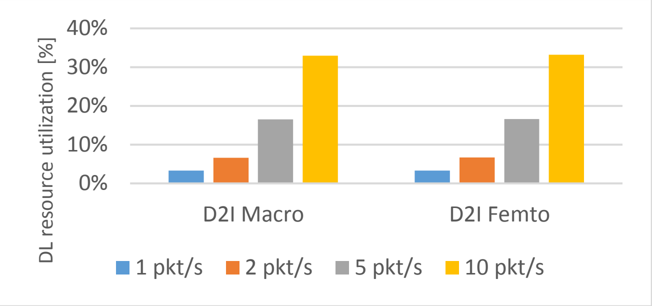

Figure 6 - DL resource utilization

V. Conclusions

In this paper we analyzed the possible solutions to bring MEC closer to IoT devices in the context of a smart-home system. First, the architecture of a smart-home IoT system based on the CoAP protocol is presented, highlighting all the network requirements, then we analyzed different network configurations that can support such systems to highlight their pros and cons. In particular, we showed the advantages of exploiting Device-to-Device (D2D) communications to guarantee proximity communication with reduced costs and better usage of resources. Finally, we demonstrated the expected benefits of each solution by means of simulations. In particular, we showed that D2D can achieve a better usage of resources for operators and lower latencies for end users.

References

[1] Hu, Y. C., et al. "Mobile Edge Computing—A Key Technology Towards 5G." *ETSI White Paper* 11 (2015).

[2] M. Patel *et al.*, “Mobile Edge Computing – Introductory Technical White Paper”, *ETSI White Paper* September 2014

[3] K. Chen, S. Lien, “Machine-to-machine communications: Technologies and challenges”, Ad Hoc Networks, Vol. 18, July 2014, pp. 3-23.

[4] F. Bonomi, R. Milito, J. Zhu, S. Addepalli. 2012. “Fog computing and

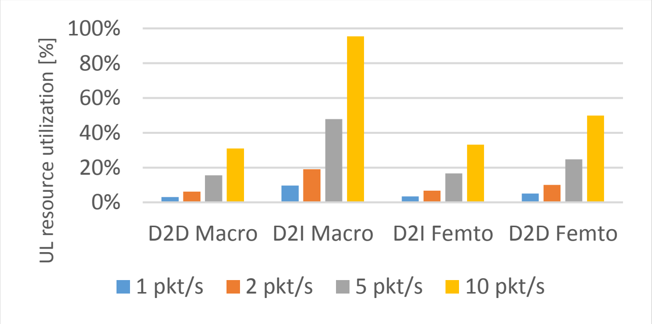

Figure 8 - UL resource utilization

its role in the internet of things”. In Proc. of 1st workshop on Mobile cloud computing (MCC '12). ACM, New York, NY, USA, 13-16.

[5] S. Andreevy, *et al.*, “Understanding the IoT Connectivity Landscape – A Contemporary M2M Radio Technology Roadmap,” IEEE Communications Magazine, accepted for publication.

[6] G. Stea, A. Virdis, "A comprehensive simulation analysis of LTE Discontinuous Reception (DRX)", Elsevier Computer Networks.

[7] 3GPP – TR 23.888 v12.0.0, “System improvements for Machine-Type Communications (MTC)(Release 12)”, March 2013

[8] C. Vallati, *et al.*, "BETaaS: A Platform for Development and Execution of Machine-to-Machine Applications in the Internet of Things", Wireless Personal Communications, 2016, vol. 87, num 8, p. 1071-1091.

[9] Z. Shelby, K. Hartke, C. Bormann, “The Constrained Application Protocol (CoAP)”. Internet RFCs, RFC 7252 (June 2014)

[10] J. Shuja, *et al.*. A Survey of Mobile Device Virtualization: Taxonomy and State of the Art. ACM Comput. Surv. 49, 1, Article 1 (April 2016).

[11] Z. Shelby, S. Krco, CoRE Resource Directory, Internet Draft, May 2012.

[12] B. Anggorojati *et al.* “BETaaS platform – a Things as a Service Environment for future M2M marketplaces”, EAI Endorsed Transactions on Cloud Systems, Vol. 1, No. 1, pp. 1-9 (February 2015)

[13] G. Nardini, G. Stea, A. Virdis, D. Sabella, M. Caretti, "Resource allocation for network-controlled device-to-device communications in LTE-Advanced", Wireless Networks, 2016.

[14] A. Asadi, Q. Wang, and V. Mancuso, (2014). A survey on device-to-device communication in cellular networks. IEEE Communications Surveys and Tutorials, 16(4), 1801–1819

[15] A. Virdis, G. Stea, G. Nardini, "Simulating LTE/LTE-Advanced Networks with SimuLTE", in “Simulation and Modeling Methodologies, Technologies and Applications”, M. S. Obaidat, T. Oeren, J. Kacprzyk, and J. Filipe, Eds., vol. 402, Springer, 2015, pp. 83–105

[16] C. Vallati, A. Virdis, E. Mingozzi, G. Stea, "Exploiting LTE D2D Communications in M2M Fog Platforms: Deployment and Practical Issues", Proc. WF-IoT 2015, Milan, Italy, 2015

[17] “OpenFog Architecture Overview”, white paper, OpenFog Consortium Architecture Working Group