

## Mobile Edge Services for Quality of Service Control and Access to Terminal Status

*Evelina Pencheva, Ivaylo Atanasov*

*Technical University of Sofia, 8 Kliment Ohridski blvd., 1000 Sofia, Bulgaria*

*E-mails: enp@tu-sofia.bg    iia@tu-sofia.bg*

**Abstract:** *An important ingredient of fifth generation (5G) networks will be Multi-access Edge Computing (MEC). MEC brings the computational intelligence of the cloud within the Radio Access Network (RAN). The virtualized functionality is accessible through Application Programming Interfaces (APIs). In this paper, we study capabilities of reuse existing time-tested Web Services to provide mobile edge middleware services. The focus is on mobile edge services that can be used by applications for bandwidth management and access to user contextual information. An extension of Web Service functionality is proposed. Implementation issues of Web Services in RAN are considered.*

**Keywords:** *Multi-access Edge Computing, Application Programming Interfaces, Radio resource control, Bandwidth management, Terminal activity.*

### 1. Introduction

5G is expected to deploy virtualized “cloud native” Radio Access Network (RAN) in order to meet high bandwidth and low-latency requirements and to enable innovative applications at the mobile edge. Multi-access Edge Computing (MEC) brings cloud capabilities into the RAN [1]. With the deployment of cloud technologies in the RAN, it is possible to reduce excessive application layer handovers and meet the performance targets for latency-critical applications [2].

MEC supports multiple access technologies such as Wi-Fi, 3G, 4G, 5G, Wi-Max, and Bluetooth. MEC can process and optimize big data collected from mobile devices before transferring them to the cloud which prevents resource wasting. It opens big opportunities for many real-time applications. Mobile edge applications are able to access directly local contextual information based on user location, local network conditions, information about users’ mobility behaviour, etc.

The European Telecommunications Standards Institute (ETSI) defined MEC reference architecture, where MEC deployment can be inside the base station or at aggregation point within Radio Access Network (RAN) [22]. Minimal latency for

many applications can be achieved by integrating MEC server inside the base station [3, 4].

MEC is a novel paradigm and it is currently under standardization. An investigation on the progress of MEC is presented in [5]. The benefits of MEC are recognized by ETSI and OpenFog Consortium which identify a number of use cases supported by MEC. The use cases include gaming, augmented and assisted reality, cognitive assistance, performance optimization, video optimization, active location tracking, etc.

MEC-service platform provides three types of middleware services: infrastructure services, radio network information services and traffic offload function [23]. Infrastructure services are used by applications for communications, service discovery and integration. Radio network information services provide authorized applications with low level real-time radio and network information related to users and cells. The traffic offload function prioritizes traffic and routes the selected, policy-based, user-data stream to and from applications that are authorized to receive the data.

The virtualized functionality is accessible through Application Programming Interfaces (APIs) and the aim is to re-use existing APIs as much as possible. In this paper, we study capabilities of reuse existing Web Services (WSs) standardized by 3GPP to provide MEC middleware services.

The Parlay X is a 3GPP standard and it defines Web Services that can invoke well defined communication functions. It enables the convergence between IP applications, WSs and telecommunications. The standard is simple and provides a high level abstraction of the underlying network and data resources. Parlay X WSs provide open access for application developers to functions in the core network. We argue that Parlay X Application-driven Quality of Service and Terminal Status may be used to provide access to functions in RAN [24], [25]. The aim of the paper is to investigate how Parlay X may provide MEC middleware services for bandwidth management and access to end-user contextual information.

The paper is structured as follows. Section 2 presents a short overview of related works. Section 3 and Section 4 present mobile edge services for quality of service control and access to information about terminal status respectively. Deployment aspects of Application-driven Quality of Service and Terminal Status in MEC environment are discussed. WSs interfaces are mapped onto network protocols. Behavioural models of the resources' state representing the network and application view are proposed. Such models are out of specifications' scope. In Section 5, some implementation details are provided as a proof of the concept. The conclusion emphasizes the authors' contribution and mentions some sketches about the functionality tests.

## 2. Related works

Several surveys focusing on MEC have been published so far [6-10]. The surveys describe MEC taxonomy and key attributes. In [6], edge computing is presented as an emerged technology which enables new capabilities including simplification of

cloud management, highly responsive services, scalability via edge analytics, privacy-policy enforcement etc. In [7], the authors describe MEC as a key ingredient of 5G and beyond systems, MEC functionality and architecture, and discusses some example use cases. Key use cases and scenarios for MEC are presented in [8], where the authors discuss standardization of MEC. The focus is on technical works dealing with computation offloading to the MEC. In [9], the current trend in MEC, its security issues/challenges and associated research challenges are enlightened. In [10], the authors make an exhaustive review on the state-of-the-art research efforts on mobile edge networks, focusing on the key enablers, such as cloud technology, network function virtualization, and smart devices.

Distributing processing logic close to the end users enables computation offloading and resource-hungry applications. An energy-efficient computation offloading scheme, which jointly optimizes offloading and radio resource allocation to obtain the minimal energy consumption under the latency constraints, is proposed in [11]. Other algorithm for the computation offloading decision weighing the energy consumption at the UE and the execution delay is proposed by [12].

Fundamental problem in distributed systems like MEC is efficiency of resource allocation. The authors of [13] formulate computation offloading decision, resource allocation and content caching strategy in MEC environment as an optimization problem, considering the total revenue of the network and solve it. Another strategy for optimizing scheduling and computational offload with low energy consumption in MEC is proposed in [14]. In [15], a novel unified resource management scheme is proposed which incorporates both centralized cloud and MEC computation offloading activities into the underlying WiFi dynamic bandwidth allocation process.

Another issue in MEC is mobility management which includes both communication handovers and computational handoff. Migration of virtual machines in MEC is costly and complicated due to heterogeneity in terminal software, MEC platform capabilities, and real-time requirements of some applications. In [16], a layered framework for active service applications is presented, where applications are encapsulated either in virtual machines or containers, which allows a substantial reduction in service downtime. The authors of [17] propose a method for reduction of handover frequency by disregarding opportunities for stronger connection with nearer base stations when the terminal experiences weaker connection with serving base station.

The review of state-of-art in MEC points out that the majority of research papers explore MEC capabilities to provide applications aimed at optimization of network resource usage and improvements of network performance. The deployment of Mobile edge applications is possible only if there is a viable MEC platform that provides middleware services. Mobile edge middleware services are provided to applications through APIs. These APIs should be vendor, product and technologies independent. As to ETSI GS MEC 002, MEC APIs should to be as simple as possible and need to response the application needs. To the extent this is possible MEC specifications have to reuse existing APIs that fulfil the requirements. In case of overloading, the access to certain APIs by an application should be dynamically controlled.

While the specifications are in process of preparation, the current state of the art in the field is covered by proprietary solutions that exhibit interoperability issues. Authors of [18] evaluate CloudAware, a context-adaptive mobile middleware for MEC that supports automated context adaptation by linking the distribution features of mobile middleware with context-aware self-adaptation techniques. An approach to design Radio Network Information WS for MEC is proposed in [19]. In [20], WS for radio resource control are presented and WS deployment issues in MEC environment are considered.

Parlay X Web Services provide standardized APIs which allow third party applications to access communication functions and information in the core network. In this paper, we evaluate the capabilities of existing Parlay X WSs to provide access to functions and information in the radio access network. For example, Parlay X Application-driven Quality of Service may be re-used to provide MEC features for bandwidth management. Moreover, Terminal Status WS forms the base for access to information about UE activity in the cell. We also propose an extension of WS functionality trying to reflect the recent advances in network evolution.

The OneAPI interfaces for quality of service control and terminal status comply with the RESTful bindings for Parlay X specifications [21]. These interfaces are a simple and elegant solution for constrained devices with limited resources. They may be used for development of fog applications.

### 3. Mobile edge service for quality of service control

#### 3.1. Parlay X support for quality of service control

The mobile edge applications for bandwidth management are generally aimed at improving performance of the network and user experience. Examples of such applications are orchestration of video streams and video optimization. Other examples include local content caching at the edge, backhaul optimization, etc.

As to the technical requirements to MEC, the mobile edge platform needs to provide a middleware service which allows authorized applications to shape selected uplink and/or downlink user plane traffic in RAN. The support of the Bandwidth Manager feature by the mobile edge platform allows a dedicated application to register its bandwidth requirements and/or priority. The mobile edge platform or a dedicated Mobile edge application may also allocate bandwidth and/or assign priority to any session or to any application.

The Application-Driven Quality of service (ADQ) is a Parlay X WS that allows applications to control the Quality of Service (QoS) available on user connection [25]. Configurable service attributes are upstream rate, downstream rate and other QoS parameters specified by the service provider. Changes in QoS may be applied either for defined time interval, or each time user connects to the network. The ADQ WS enables applications to register with the service for notifications about network events that affect QoS, temporary configured on the user's connection. Currently, the ADQ WS does not support functions for usage monitoring, session termination and application detection and control.

Using ADQ interfaces an authorized application may apply temporary QoS parameters, change QoS parameters, and release QoS parameters. The application can manage QoS event registrations. By using information about bearers the application may improve service execution. For example, the application may initiate session release on behalf of the user after indication that all radio access bearers assigned to the ongoing session are released, but the core network has not received session termination request from the UE itself.

The mobile edge platform should allow the collection of charging-related information such as traffic usage, application instantiation, access, usage duration, resource usage, etc. An authorized application may be interested in the accumulated usage of network resources on per IP session and per user basis. This capability may be required for applying QoS control based on the total network usage in real-time. For example, the application may change the charging rate based on the resource usage (e.g., applying discounts after a specified volume is reached). Another usage is the assignment of a common quota for mobile accesses for a limited time period for a defined set of subscriptions. The base station(s) monitors the common quota during each session which may be consumed by one or more UEs over the wireless network. When a defined percentage of the common quota and/or all common quotas have been consumed, the application may be notified of the event. In this case, the application may prevent the access to the services. Further, the application may inspect specific application traffic.

In order to provide usage monitoring, an authorized application should be able to set the applicable thresholds for usage monitoring, to start and stop the usage monitoring. The usage monitoring, if activated, may be performed for a particular application, a group of applications or all detected traffic within a specific multimedia session. The application should be notified when the provided usage monitoring thresholds have been reached. The ADQ WS does not provide usage monitoring functions.

An authorized application may request the detection of specified application traffic and notifications on start or stop of application traffic. The enforcement actions may include blocking of the application traffic, bandwidth limitations, or redirection of traffic to another address. For example, the application detection and control functionality may be used to detect traffic of point-to-point file sharing protocol, where the application may limit the bit rate, used by the service.

In order to provide application detection and control, an authorized edge application should be able to set inspection for specific application traffic. The application should be notified on traffic start and stop, and be able to apply the specified enforcement actions. The ADQ WS does not provide application detection and control functions.

### 3.2. Mapping of ADQ interfaces onto network protocols

The 4G RAN is composed of one network element: the eNodeB (base station). ENodeB functionalities related to QoS include bearer level rate enforcement, bearer level admission control and congestion control. The protocol between UE and eNodeB is Radio Resource Control (RRC) [26]. The main services related to QoS

control include establishment, maintenance and release of an RRC connection between the UE and radio access network (E-UTRAN), establishment, configuration, maintenance and release of point to point radio bearers, and QoS management functions. The interfaces between eNodeB and Mobility Management Entity in the core network is named S1 and the respective control protocol is S1 Application Protocol (S1AP) [27]. Among the others, S1AP protocol is in charge of E-UTRAN Radio Access Bearers (E-RABs) management.

Deployment of ADQ WS in MEC environment requires mapping of ADQ APIs onto S1AP procedures.

The ApplicationQoS interface provides operations for QoS control. The invocation of interface operations results in initiation of S1AP procedures related to E-RABs. The applyQoSFeature operation is used by application to request a temporary QoS feature to be setup on the end user connection. Implemented in MEC platform, this operation allows the application to assign resources on air interface and S1 interfaces for one or several E-RABs and to setup corresponding data radio bearers for given UE. The operation initiates E-RAB Setup procedure. The Parlay X ADQ defines only DownStreamSpeedRate and UpStreamSpeedRate as QoS attributes. We suggest an extension of QoSFeatureProperties structure with elements indicating QoS parameters of bearers in Evolved Packet System, namely QoS Class Identifier, Allocation and Retention Priority (ARP), Guaranteed Bit Rate (GBR) and Maximum Bit Rate for GBR bearers and APN Aggregated Maximum Bit Rate (APN-AMBR) and UE-AMBR. Fig. 1 shows the XML scheme related to E-RAB QoS parameters.

When the eNodeB reports unsuccessful establishment of an E-RAB, the cause value indicates the reason for such unsuccessful establishment, e.g., “Radio resources not available”, “Failure in the Radio Interface Procedure”. The respective WS exception is “Insufficient connection resources”. If the eNodeB receives E-RAB Setup Request message containing incorrect E-RAB ID or QoS parameter values indicating QCI of GBR bearer, which does not contain GBR QoS information, then the eNodeB considers the establishment of the corresponding E-RAB as failed and the WS generates exception “Invalid input value”.

The ADQ modifyQoSFeature operation may be used by an application to alter the QoS attributes of an active temporary QoS feature. This operation initiates E-RAB Modification procedure. The service exception “Invalid input value” rises when the eNodeB finds an incorrect E-RAB ID. The removeQoSFeature operation may be used by the application to release a temporary QoS value, which is currently active on the user session. This operation initiates E-RAB Release procedure.

The getQoSStatus operation falls in the category of self-care and may be used to retrieve the status of an end user connection. The response to this method will contain information about the characteristics of the active E-RABs including information about the temporary QoS features that are currently active on the end user connection.

ADQ provides interfaces which may be used by a Mobile edge application to manage its registration for notifications. When eNodeB sends an indication that one or several E-RABs for a UE are released or modified, the application is also notified in case of active subscription.

```

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://example.com" targetNameSpace="http://example.com">
  <xs:simpleType name="bitRate">
    <xs:restriction base="xs:nonNegativeInteger">
      <xs:maxInclusive value="1000000000"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="QoSClassIndicator">
    <xs:restriction base="xs:nonNegativeInteger">
      <xs:maxInclusive value="255"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="PriorityLevel">
    <xs:restriction base="xs:nonNegativeInteger">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="15"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="PreemptionCapability">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ShallNotTriggerPreemption"/>
      <xs:enumeration value="MayTriggerPreemption"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="PreemptionVulnerability">
    <xs:restriction base="xs:string">
      <xs:enumeration value="NotPreemptable"/>
      <xs:enumeration value="Preemptable"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="AllocationAndRetentionPriority">
    <xs:sequence>
      <xs:element name="PL" type="PriorityLevel"/>
      <xs:element name="PC" type="PreemptionCapability"/>
      <xs:element name="PV" type="PreemptionVulnerability"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="GuaranteedBitRate">
    <xs:sequence>
      <xs:element name="ERABMaxBitRateDL" type="bitRate"/>
      <xs:element name="ERABMaxBitRateUL" type="bitRate"/>
      <xs:element name="ERABGuaranteedBitRateDL" type="bitRate"/>
      <xs:element name="ERABGuaranteedBitRateUL" type="bitRate"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="UEAggregateBitRate">
    <xs:sequence>
      <xs:element name="UEAggregateBitRateDL" type="bitRate"/>
      <xs:element name="UEAggregateBitRateUL" type="bitRate"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="ERABLevelQoSParameters">
    <xs:sequence>
      <xs:element name="QCI" type="QoSClassIndicator"/>
      <xs:element name="ARP" type="AllocationAndRetentionPriority"/>
      <xs:element name="GBR" type="GuaranteedBitRate" minOccurs="0" maxOccurs="1"/>
      <xs:element name="ABR" type="UEAggregateMaxBitRate" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>

```

Fig. 1. XML scheme related to E-RAB QoS parameters

Fig. 2 shows the sequence for application control on QoS.

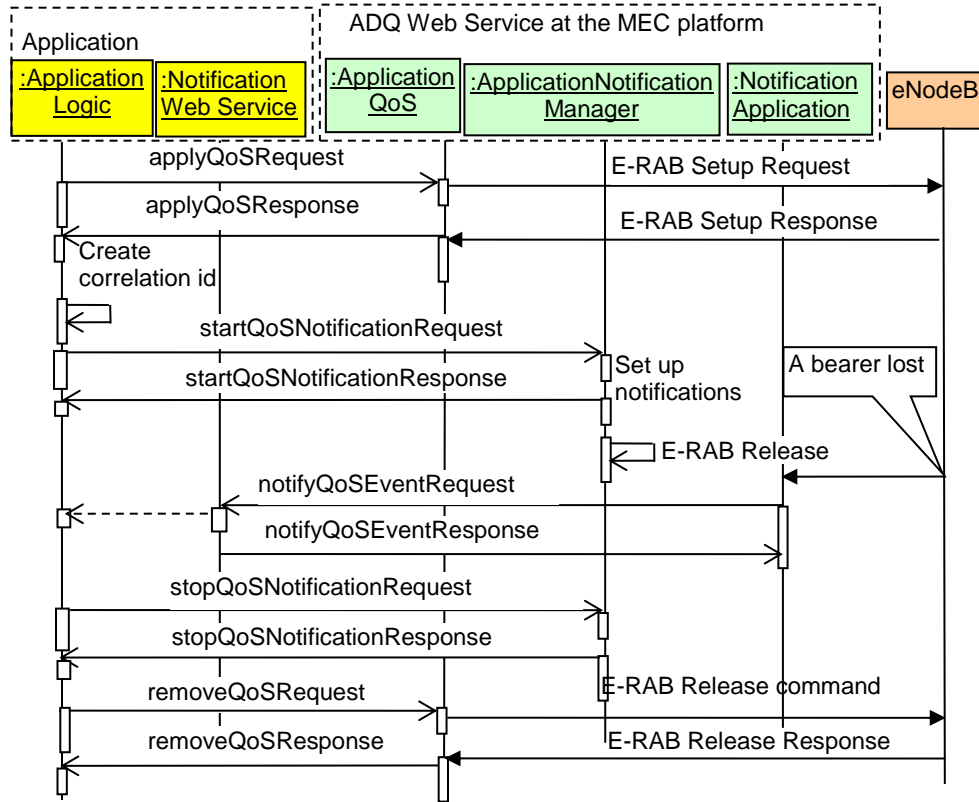


Fig. 2. Application control on QoS available on the UE session

The mobile edge application indicates its interest in receiving notifications by registering for events. When an event occurs in the network that merits a notification to be raised, interested application receives a notification and its implications on the temporary QoS features active on the end user connection.

The message sequence in the Fig. 2 may be interpreted in the following context. The Streaming Application makes a request to the ADQ WS to apply specific QoS parameter values on existing UE connection. Assuming that the network supports the requested QoS values, the new QoS is applied as requested by the application. The application subscribes for notifications. When a bearer is lost (e.g., due to bad radio conditions), the application is notified. The application stops the notifications and requests release of the UE session.

### 3.3. Resource state models

As a mediation point between mobile edge applications and RAN, the mobile edge platform, which provides Web Services, needs to maintain the network and application views on the status of the applied QoS resources. These views need to be synchronized.



Fig. 3 shows the proposed simplified state model of QoS resources assigned for a user connection that has to be maintained by the eNodeB. When the user connects to the network, an RRC connection and a default E-RAB are established. A user connection with specific QoS requires establishment of a dedicated E-RAB with appropriate QoS, which results in RRC connection re-configuration. The transitions in the state model correspond to the respective RRC and S1AP messages.

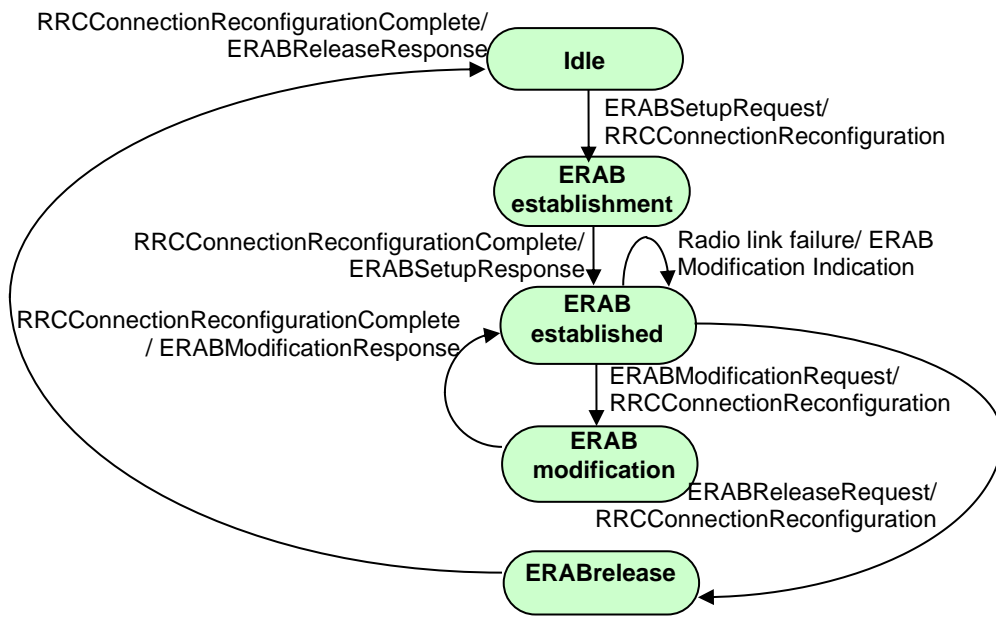


Fig. 3. State model of QoS resources maintained by the eNodeB

Fig. 4 shows the proposed simplified state model of QoS resources as seen by a Mobile edge application. The transitions in the state model correspond to the triggers for QoS control and WS messages.

Both models are synchronized as they expose equivalent behaviour in consequence of events related to QoS control, i.e. for applying specific QoS control by a mobile edge application, the network establishes the respective E-RABs, and for modifying, or deleting specific QoS features by a mobile edge application, the network modifies, or deletes the respective E-RABs.

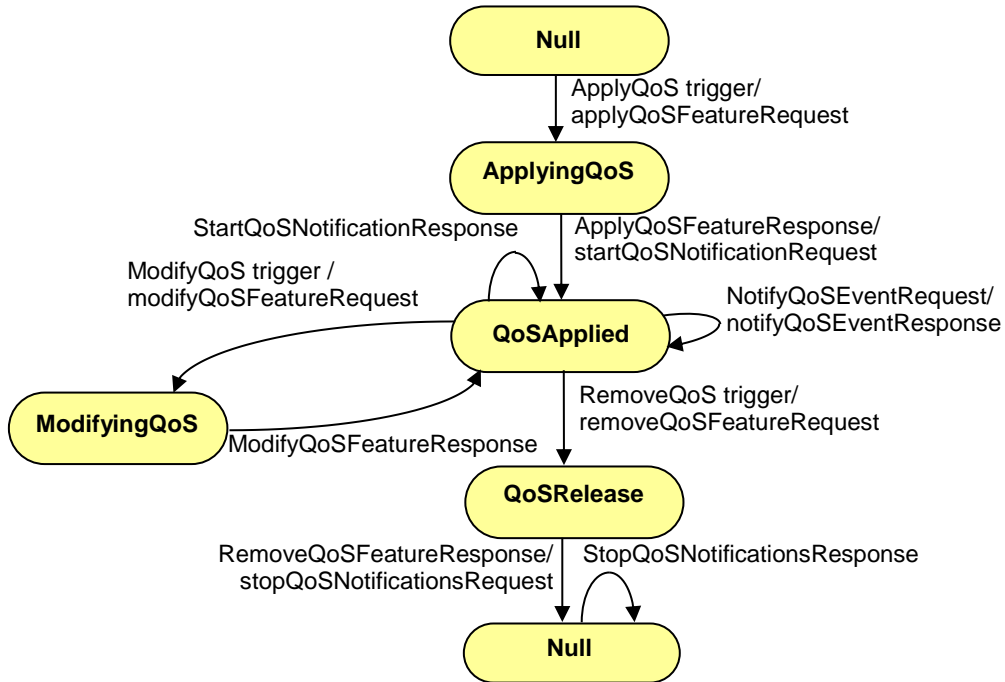


Fig. 4. State model of QoS resources maintained by mobile edge application

### 3.4. Extending the functionality of Parlay X ADQ WS

The Parlay X ADQ WS may be extended in order to support usage monitoring and application detection and control. The usage monitoring functionality allows a mobile edge application to set applicable thresholds for the accumulated usage of network resources. The application detection and control functionality comprises the request to detect the specified application traffic, reports to a mobile edge application on start or stop of application traffic, and to apply the specified enforcement and charging actions.

Fig. 5 summarizes the proposed interfaces for usage monitoring.

The UsageMonitoringManager interface provides operations for management of monitoring on bearer resource usage. The interface provides methods that allow an authorized mobile edge application to set usage monitoring thresholds (setUsageMonitoringThresholds operation), to start (startUsageMonitoring operation), and to stop usage monitoring (stopUsageMonitoring operation). The application may decide to release the user session (sessionTermination operation) when the thresholds are reached or user credit is over. The UsageNotification interface allows the application to monitor the bearer usage. The usageThresholdReached operation reports the usage thresholds parameters set by the applications have been reached. The usageError operation sends an error message to the multi access edge application to indicate that the notification for resource usage is cancelled by the WS. The notificationEnd operation informs the mobile edge

application that the notifications have been completed when the duration expires or the limit for notifications has been reached.

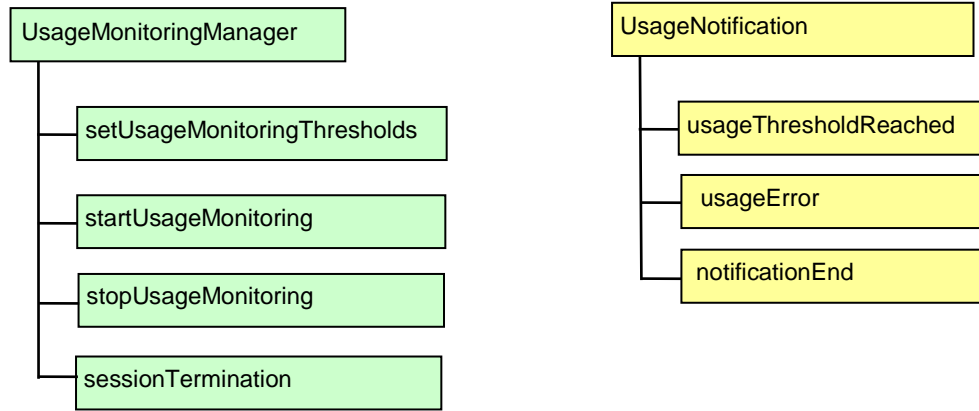


Fig. 5. Usage monitoring MEC interfaces

Following the same approach interfaces for periodic usage monitoring may be defined. An example of WSDL description is given in Section 6.

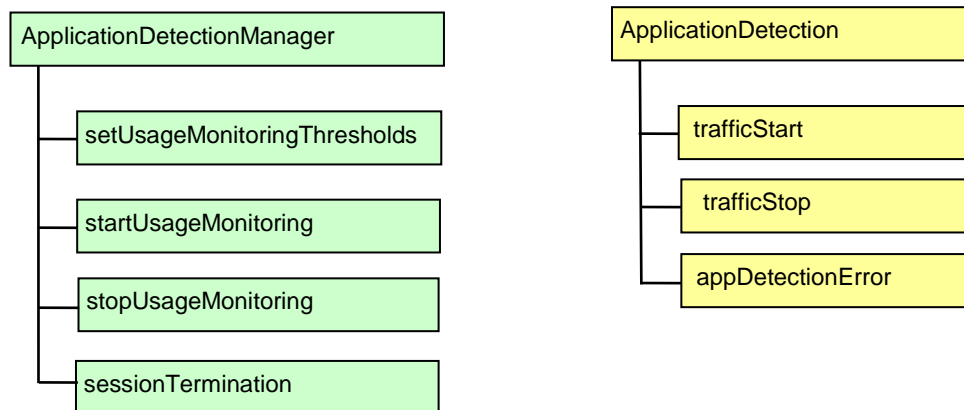


Fig. 6. MEC interfaces for application detection and control

The **ApplicationDetectionManager** interface provides operations that allow a Mobile edge application to register its interest in detection of specific applications. The interface provides methods that allow an authorized application to start and stop inspection of specific application traffic. The **ApplicationDetection** interface allows a Mobile edge application to receive notifications about the start and stop of application traffic events and to indicate how the traffic should be handled. The `trafficStart` operation reports the start of monitored traffic and provides instructions how the traffic should be handled. Possible actions are “continue”, “change of QoS” for the application traffic, which results in E-RAB modification, or “block” the application traffic which results in E-RAB release. The `applicationDetectionError`

operation sends an error message to the Mobile edge application to indicate the notification for application detection is cancelled by the WS. Fig. 6 summarizes the proposed interfaces for application detection and control.

## 4. Mobile edge service for access to terminal status

### 4.1. Parlay X support for access to information about terminal status

Mobile edge platform may provide up-to-date radio network information which can be generated locally or be based on information received from external sources. The radio network information may be provided at the relevant granularity (e.g., per UE or per cell, per period of time) [24]. The UE related information includes information about UEs connected to the eNodeB(s), their UE context and the related radio access bearers, as well as information about changes in their UE context and radio access bearers.

As to [25], the Parlay X Terminal Status WS provides access to the status of a terminal through:

- request for the status of a terminal;
- request for the status of a group of terminals;
- notification of a change in the status of a terminal.

It may be used to provide information about UE activity in a cell. UE status in a cell can be expressed as not serviced, active or inactive. The terminal is in not serviced state if the UE is not camped at the cell and the eNodeB does not serve it. The terminal is in active state if it is camped at the cell and involved in a session. The terminal is in inactive state if it is camped at the cell but it is not involved in a session. When a request for the activity of a group of terminals in a cell is made, the response may contain a full or partial set of results. This allows the service to provide results based on a number of criteria including number of terminals for which the request is made and amount of time required to retrieve the information. The application may initiate additional requests for terminals with incomplete information.

The terminal activity in a cell may be interesting for different applications such as radio access bearer monitoring, resolving network congestion, etc. An example is an analytic application that monitors the changes in the activity of a group of terminals (e.g., from active to idle and back to active) in order to predict the usage of cell resources and optimize resource scheduling. Another example is an application that resolves specific situations of network congestion in a dense environment. In case of network congestion, the application can communicate with counterpart applications running on devices, to request them to activate direct device-to-device communication through application-specific means.

The information about terminal activity in a cell in MEC environment may be provided on the base of UE context status in the eNodeB. UE context stored in the eNodeB contains information about signalling connection ID between the Mobility Management Entity (MME) in the core network and UE, security context, roaming and access restrictions, UE capability information and bearer setup information.

The S1AP protocol has functions related to UE context management. The UE context management functions include Initial context transfer, UE context release, UE context modification and UE context resumption. Initial context transfer functionality is used to establish an S1UE context in the eNodeB, to setup the default IP connectivity, to setup one or more E-RABs if requested by the MME, and to transfer non access stratum signalling related information to the eNB if needed. UE context modification functionality allows changing the established UE context partly. UE context resumption functionality allows keeping the UE Context in the eNodeB for a UE which does not have a RRC connection with the eNodeB and to resume the RRC connection without the need to re-establish the UE Context.

#### 4.2. Resource state models

The mobile edge platform needs to maintain in a synchronized way the models representing the application view on the terminal state and the UE context state as seen by the eNodeB.

Fig. 7 shows the UE context state model. In Null state, there is no UE context for the terminal in the eNodeB, i.e. the terminal is not served by the eNodeB. In Established state, there exists a UE context for the terminal in the eNodeB and the terminal may send or received data using the established data bearers. In Suspended state, the UE context is suspended e.g. due to terminal inactivity. The transitions in the UE context state model are triggered by S1AP procedures.

The Initial Context Setup procedure is initiated by the MME. The purpose of the Initial Context Setup procedure is to establish the necessary overall initial UE Context including E-RAB context, the security key, handover restriction list, UE radio capability and UE security capabilities etc. In case of successful initial context establishment, this procedure leads to establishment of radio bearer between the UE and eNodeB. The UE may use the established radio bearers to transmit user data. The Initial Context Setup procedure may fail, e.g., because the encryption or integrity protection algorithm in the UE security capabilities do not match any allowed algorithms defined in the eNodeB.

The UE Context Modification procedure enables the MME to partly modify the UE context in the eNodeB (e.g., security key or aggregated maximum bit rate) for UEs in active state. It is initiated by the MME. The procedure may fail if the eNodeB receives both the circuit switched fallback indicator and one of the security parameters.

The eNodeB may initiate UE Context Suspend procedure due to UE inactivity if the UE and the network support signalling optimizations to enable efficient transport of user data over the user plane failed. Data related to the S1AP association, UE Context and bearer context, necessary to resume the connection are kept in the eNodeB, UE and the MME.

The purpose of the UE Context Resume procedure is to indicate to the MME that the UE has resumed the suspended RRC connection and to request the MME to resume the UE context, UE-associated logical S1-connection and the related bearer contexts in the core network.

If the UE remains inactive for certain amount of time, the eNodeB initiates UE context release. Another reasons for eNodeB initiated UE context release may be loss of radio connection with UE, subscription expiry of a closed subscriber group, circuit switched fallback trigger, intersystem redirection and UE unavailability for packet switched services. UE context release may be initiated also by the MME in case of requirements for load balancing and offload. This procedure leads to release of all related signalling and user data transport resources in the eNodeB.

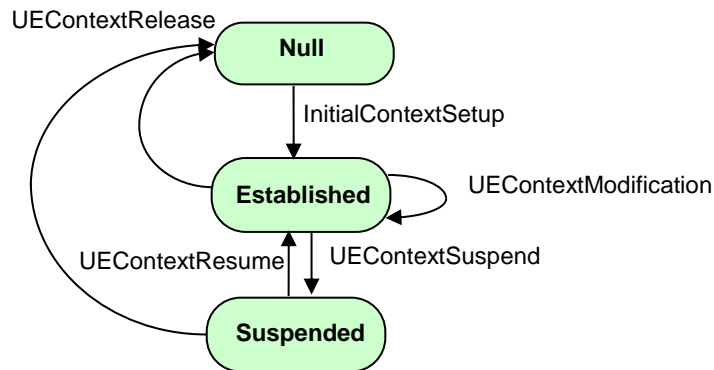


Fig. 7. State model of UE context in eNB

Fig. 8 shows the terminal state models as seen by the application. In NotServiced state, the terminal is not registered in the cell and it cannot participate in data communications in the cell. In Active state, the terminal participates in data communication. In Inactive state, the terminal is serviced by the eNodeB, but currently is not involved in data communications. The state transitions are triggered by WS notifications.

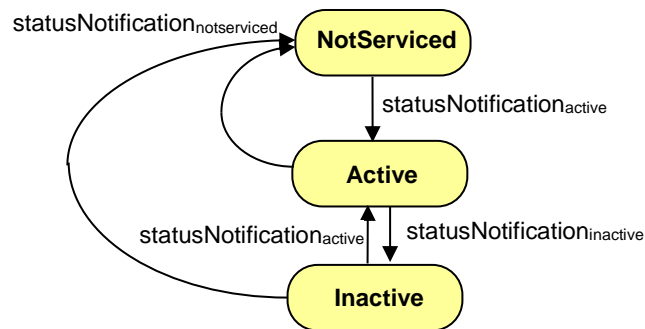


Fig. 8. Application view on the terminal intra-cell activity

Both models expose equivalent behaviour as each of the procedures related to UE context management result in a respective notification to interested mobile edge applications.

## 5. Proof of the concept

The proof of the concept has its significance regarding the validation goals in aspect of the approach's implementation.

Looked at from topology point of view, the proof is conducted by using of front-/back-end approach, which is applied to two couples of Virtual Machines (VMs) in lab environment. The reason that much simpler single-node type of proof is considered inappropriate is that it lacks the scalability component, which is seen as “conditio sine qua non”. Another argument is that the incurred latency is less realistic, which might get critical when dealing with finite state machines, or the earlier eventual latency issues become evident, the better. The VMs form in-memory grid that helps for service integration by its message-passing bus which brings at least twofold gain – keeps the scalability feature intact and makes the dialogues between state machines easier to monitor.

```
</wsdl:message>
<wsdl:message name="startPeriodicUsageMonResponse">
  <wsdl:part element="tns:startPeriodicUsageMonResponse" name="startPeriodicUsageMonResponse"> </wsdl:part>
</wsdl:message>
<wsdl:portType name="SubscriptionsPort">
  <wsdl:operation name="stopPeriodicUsageMon">
    <wsdl:input message="tns:stopPeriodicUsageMonRequest" name="stopPeriodicUsageMonRequest"> </wsdl:input>
    <wsdl:output message="tns:stopPeriodicUsageMonResponse" name="stopPeriodicUsageMonResponse"> </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="startPeriodicUsageMon">
    <wsdl:input message="tns:startPeriodicUsageMonRequest" name="startPeriodicUsageMonRequest"> </wsdl:input>
    <wsdl:output message="tns:startPeriodicUsageMonResponse" name="startPeriodicUsageMonResponse"> </wsdl:output>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="SubscriptionsPortSoap11" type="tns:SubscriptionsPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="stopPeriodicUsageMon">
    <soap:operation soapAction="">
      <wsdl:input name="stopPeriodicUsageMonRequest">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="stopPeriodicUsageMonResponse">
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  <wsdl:operation name="startPeriodicUsageMon">
    <soap:operation soapAction="">
      <wsdl:input name="startPeriodicUsageMonRequest">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="startPeriodicUsageMonResponse">
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:service>
</wsdl:definitions>
<!-- SOAP Envelope for startPeriodicUsageMonRequest -->
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header/>
  <soap:Body>
    <bs:startPeriodicUsageMonRequest xmlns:bs="http://schemas.xmlsoap.org/soap/envelope/">
      <bs:interfaceName>nc-ws</bs:interfaceName>
      <bs:correlator>df45-0013</bs:correlator>
      <bs:reference/>
      <bs:requester/>
      <bs:addresses/>
      <bs:frequency>
        <bs:metric>Second</bs:metric>
        <bs:unit>15</bs:unit>
      </bs:frequency>
      <bs:duration>
        <bs:metric>Hour</bs:metric>
        <bs:unit>1</bs:unit>
      </bs:duration>
    </bs:startPeriodicUsageMonRequest>
  </soap:Body>
</soap:Envelope>
<!-- SOAP Envelope for stopPeriodicUsageMonRequest -->
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header/>
  <soap:Body>
    <bs:stopPeriodicUsageMonRequest xmlns:bs="http://schemas.xmlsoap.org/soap/envelope/">
      <bs:correlator>2301</bs:correlator>
    </bs:stopPeriodicUsageMonRequest>
  </soap:Body>
</soap:Envelope>
```

Fig. 9. A part of the contract definition, given in WSDL that realizes the periodic usage monitoring notifications

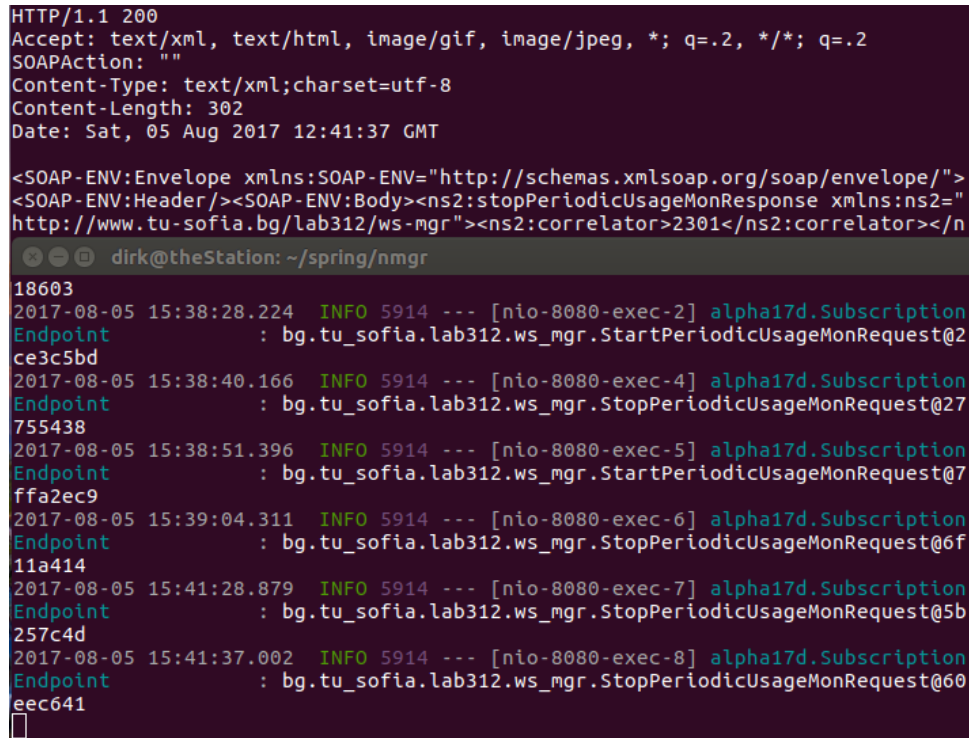
In order to show the viability of the approach, it is shown in Fig. 9 a part of the contract definition, given in WSDL that realizes the periodic usage monitoring notifications with the typical start/stop couple of operations. The preferred version of the SOAP protocol is 1.1 as far as it gives a wider deployment base eventually. The SOAP envelopes for both types of operations to be depicted. The starting one (on left) shows a setup of notification subscription for an hour with frequency of 15 s, while the second (on right) is effectively stopping another subscription. The response message parts of the operations are not shown in Fig. 9 to keep it more readable.

The message exchange shown in Fig.10 is taken on the service interface between the service endpoint and a client. The transport, i.e., the HTTP code/headers, of the response message of the stop operation and the SOAP/domain part are given, while at the bottom is the service activity log.

```

HTTP/1.1 200
Accept: text/xml, text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
SOAPAction: ""
Content-Type: text/xml; charset=utf-8
Content-Length: 302
Date: Sat, 05 Aug 2017 12:41:37 GMT

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/><SOAP-ENV:Body><ns2:stopPeriodicUsageMonResponse xmlns:ns2="
http://www.tu-sofia.bg/lab312/ws_mgr"><ns2:correlator>2301</ns2:correlator></n
  
```



```

dirk@theStation: ~/spring/nmgr
18603
2017-08-05 15:38:28.224 INFO 5914 --- [nio-8080-exec-2] alpha17d.Subscription
Endpoint : bg.tu_sofia.lab312.ws_mgr.StartPeriodicUsageMonRequest@2
ce3c5bd
2017-08-05 15:38:40.166 INFO 5914 --- [nio-8080-exec-4] alpha17d.Subscription
Endpoint : bg.tu_sofia.lab312.ws_mgr.StopPeriodicUsageMonRequest@27
755438
2017-08-05 15:38:51.396 INFO 5914 --- [nio-8080-exec-5] alpha17d.Subscription
Endpoint : bg.tu_sofia.lab312.ws_mgr.StartPeriodicUsageMonRequest@7
ffa2ec9
2017-08-05 15:39:04.311 INFO 5914 --- [nio-8080-exec-6] alpha17d.Subscription
Endpoint : bg.tu_sofia.lab312.ws_mgr.StopPeriodicUsageMonRequest@6f
11a414
2017-08-05 15:41:28.879 INFO 5914 --- [nio-8080-exec-7] alpha17d.Subscription
Endpoint : bg.tu_sofia.lab312.ws_mgr.StopPeriodicUsageMonRequest@5b
257c4d
2017-08-05 15:41:37.002 INFO 5914 --- [nio-8080-exec-8] alpha17d.Subscription
Endpoint : bg.tu_sofia.lab312.ws_mgr.StopPeriodicUsageMonRequest@60
e6c641
  
```

Fig. 10. Snapshot of message exchange and activity log

## 6. Conclusion

A prerequisite for the wide deployment of innovative applications with high bandwidth and low latency requirements in the radio access network is the availability of viable, scalable and resilient mobile edge platform. The mobile edge platform APIs should be application agnostic. The APIs need to promote application interoperability and portability on different mobile edge platforms.

In this paper, we investigate the extent, to which existing APIs that fulfil the requirements could be reused. The research covers established and time-tested Web Services that provide abstracted access to communication functions and information in the core network. The focus is on Application-driven Quality of Service and Terminal Status Web Services. Application-driven Quality of Service may be used to provide the Bandwidth Manager feature, which allows an authorized application to register statically and/or dynamically its bandwidth requirements and/or priority, as well as to allocate bandwidth and/or assign priority to any session or to any application. Terminal Status WS may be used to implement the Radio Network



Information feature, which provides information about UEs activity in a cell. Implementation of mobile edge platform requires identification of homomorphism between WS operations and network control protocol messages. Such homomorphism is identified for operations of ADQ and Terminal Status WSs and messages of the protocol S1AP. The control logic at the mobile edge platform side and application side is modelled. An extension of ADQ WS is proposed in order to face the requirements for usage monitoring and application detection and control in RAN.

The approach is flexible and open to further extensions toward RESTful-type of solutions by wrapping the front-end with appropriate adapters.

Re-use of existing API provides a simple and controllable access to functions in the radio access network. For third party application developers it is an opportunity to create new attractive applications, and for network operators it is a new source of revenue generation due to the reduction of data delivery cost.

**Acknowledgements:** The research is conducted under the grant of project DH07/10-2016, funded by National Science Fund, Ministry of Education and Science, Bulgaria.

## References

1. Taleb, T., K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Architecture & Orchestration. – IEEE Communications Surveys & Tutorials, Vol. **PP**, 2017, No 99.
2. Pham, N. M. N., V. S. Le, H. H. C. Nguyen. Energy Efficient Resource Allocation for Virtual Services Based on Heterogeneous Shared Hosting Platforms in Cloud Computing. – Cybernetics and Information Technologies, Vol. **17**, 2017, No 3, pp. 47-57.
3. Ahmed, A., E. Ahmed. A Survey on Mobile Edge Computing. – In: 10th International Conference on Intelligent Systems and Control (ISCO'16), Coimbatore, 2016, pp. 1-8.
4. Jararweh, Y., M. Alsmirat, M. Al-Ayyoub, E. Benkhelifa, A. Darabseh, B. Gupta, A. Doulat. Software-Defined System Support for Enabling Ubiquitous Mobile Edge Computing. – The Computer Journal, February 2017, pp. 1-15.
5. Li, H., G. Shou, Y. Hu, Z. Guo. Mobile Edge Computing: Progress and Challenges. – In: Proc. of 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud'16), Oxford, 2016, pp. 83-84.
6. Satyanarayanan, M. The Emergence of Edge Computing. – Computer, January 2017, pp. 30-39.
7. Lian, B. Mobile Edge Computing. Book Chapter. – In: Key Technologies for 5G Wireless Systems. V. W. S. Wong, R. Schober, D. W. K. Ng, L.-C. Wang, Eds. Cambridge University Press, 2017.
8. Mach, P., Z. Becvar. Mobile Edge Computing: A Survey on Architecture and Computation Offloading. Cornell University Library, arXiv:1702.05309v2 [cs.IT]
9. Kumar, G. K., M. Gobi. Survey on Mobile Cloud Computing [MCC], its Security & Future Research Challenges. – International Research Journal of Engineering and Technology (IRJET), Vol. **4**, June 2017, Issue 6, pp. 2987-2995.
10. Wang, S., X. Zhang, Y. Zhang, L. Wang, J. Yang, W. Wang. A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications. – IEEE Access, Vol. **5**, 2017, pp. 6757-6779.
11. Zhang, K., Y. Mao, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, Y. Zhang. Energy-Efficient Offloading for Mobile Edge Computing in 5G Heterogeneous Networks. – In: IEEE Access, January 2016, pp. 1-10.
12. Chen, X., L. Jiao, W. Li, X. Fu. Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing. – IEEE/ACM Transactions on Networking, Vol. **24**, October 2016, No 5, pp. 2795-2808.

13. Wang, C., C. Liang, F. R. Yu, Q. Chen, L. Tang. Computation Offloading and Resource Allocation in Wireless Cellular Networks with Mobile Edge Computing. – IEEE Transactions on Wireless Communications, Vol. **PP**, 2017, No 99.
14. Labidi, W., M. Sarkiss, M. Kamoun. Joint Multi-User Resource Scheduling and Computation Offloading in Small Cell Networks. – In: IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'15), 2015, pp. 794-801.
15. Rimal, B. P., D. P. Van, M. Maier. Mobile-Edge Computing vs. Centralized Cloud Computing over a Converged FiWi Access Network. – IEEE Transactions on Network and Service Management, Vol. **PP**, 2017, No 99.
16. Machen, A., S. Wang, K. K. Leung, B. J. Ko, T. Salonidis. Live Service Migration in Mobile Edge Clouds. Cornell University Library, arXiv:1706.04118 [cs.DC], 2017.
17. Bao, W., B. Liang. Stochastic Geometric Analysis of User Mobility in Heterogeneous Wireless Networks. – IEEE Journal on Selected Areas in Communications, Vol. **33**, 2015, No 10, pp. 2212-2225.
18. Orsini, G., D. Bade, W. Lamersdorf. CloudAware: A Context-Adaptive Middleware for Mobile Edge and Cloud Computing Applications. – In: IEEE 1st International Workshops on Foundations and Applications of Self\* Systems (FAS\*W'16), Augsburg, 2016, pp. 216-221.
19. Pencheva, E., I. Atanasov. An Approach to Design Radio Network Information Web Services for Mobile Edge Computing. – In: IEEE Global IoT Summit, Geneva, Switzerland, 2017, pp. 1-6.
20. Atanasov, I., E. Pencheva. Web Services for Radio Resource Control. – In: 2nd EAI International Conference on Cloud, Networking for IoT Systems, Brindisi, Italy, 2017, pp. 1-6.
21. Jakobsson, S., C. Mulligan, M. Unmehopa. Research and Reality: The Evolution of Open Network API Standards. – In: IEEE International Conference on Communications (ICC'12), Ottawa, ON, 2012, pp. 6901-6905.
22. ETSI GS MEC 003. Mobile Edge Computing (MEC). Framework and Reference Architecture. V1.1.1. 2016.
23. ETSI GS MEC 002. Mobile Edge Computing (MEC). Technical Requirements. V1.1.1. 2016.
24. 3GPP TS 29.199-17. Technical Specification Group Core Network and Terminals. Open Service Access (OSA). Parlay X Web Services. Part 17: Application-Driven Quality of Service. Release 9. V9.0.0. 2009.
25. 3GPP TS 29.199-8. Technical Specification Group Core Network and Terminals. Open Service Access (OSA). Parlay X Web Services. Part 8: Terminal Status. Release 9. V9.0.0. 2009.
26. 3GPP TS 36.331. Evolved Universal Terrestrial Radio Access (EUTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN). Radio Resource Control (RRC). Protocol Specification. Release 14. V14.1.0. 2016.
27. 3GPP TS 36.413. Evolved Universal Terrestrial Radio Access (EUTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN). S1 Application Protocol (S1AP). Release 14. V14.1.0. 2017.

*Received 02.11.2017; Second Version 05.03.2018; Accepted 16.03.2018*